NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

QUANTIFYING THE DIFFERENCES IN LOW PROBABILITY OF INTERCEPT RADAR WAVEFORMS USING QUADRATURE MIRROR FILTERING

by

Pedro Jarpa

September 2002

Thesis Advisor:	Phillip E. Pace
Co - Advisor:	Herschel H. Loomis, Jr.

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY	**2. REPORT DATE** September 2002	**3. REPORT TYPE AND DATES COVERED** Electrical Engineer's Thesis	
4. TITLE AND SUBTITLE: Quantifying the Differences in Low Probability of Intercept Radar Waveforms Using Quadrature Mirror Filtering			**5. FUNDING NUMBERS**
6. AUTHOR(S) Jarpa, Pedro			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT

Low Probability of Intercept (LPI) radars are a class of radar systems that possess certain performance characteristics causing them to be nearly undetectable by most modern digital intercept receivers. Consequently, LPI radar systems can operate undetected until the intercept receiver is much closer than the radar's target detector. The enemy is thus faced with a significant problem. To detect these types of radar, new direct digital receivers that use sophisticated signal processing are required. This thesis describes a novel signal processing architecture, and shows simulation results for a number of LPI waveforms. The LPI signal detection receiver is based on Quadrature Mirror Filter Bank (QMFB) Tree processing and orthogonal wavelet techniques to decompose the input waveform into components representing the signal energy in rectangular "tiles" in the time-frequency plane. By analyzing the outputs at different layers of the tree it is possible to do feature extraction, identify and classify the LPI waveform parameters, and distinguish among the various LPI signal modulations. Waveforms used as input signals to the detection algorithm include Frequency Modulated Continuous Wave, Polyphase Codes, Costas Codes and Frequency Shift Keying/Phase Shift Keying waveforms. The output matrices resulting from the most relevant layers of the QMFB tree processing are examined and the LPI modulation parameters are extracted under various signal-to-noise ratios.

14. SUBJECT TERMS Signal Processing, Digital Filters, LPI, LPI Radar Signals, Quadrature Mirror Filter Bank.			**15. NUMBER OF PAGES** 174
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

QUANTIFYING THE DIFFERENCES IN LOW PROBABILITY OF INTERCEPT RADAR WAVEFORMS USING QUADRATURE MIRROR FILTERING

Pedro F. Jarpa
Captain, Chilean Air Force
B.S., Military Polytechnic Academy, Chilean Army 1998

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
September 2002

Author: Pedro F. Jarpa.

Approved by: Phillip E. Pace, Thesis Advisor.

Herschel H. Loomis, Jr., Co-Advisor.

John P. Powers, Chairman
Department of Electrical and Computer Engineering.

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Low Probability of Intercept (LPI) radars are a class of radar systems that possess certain performance characteristics causing them to be nearly undetectable by most modern digital intercept receivers. Consequently, LPI radar systems can operate undetected until the intercept receiver is much closer than the radar's target detector. The enemy is thus faced with a significant problem. To detect these types of radar, new direct digital receivers that use sophisticated signal processing are required. This thesis describes a novel signal processing architecture, and shows simulation results for a number of LPI waveforms. The LPI signal detection receiver is based on Quadrature Mirror Filter Bank (QMFB) Tree processing and orthogonal wavelet techniques to decompose the input waveform into components representing the signal energy in rectangular "tiles" in the time-frequency plane. By analyzing the outputs at different layers of the tree it is possible to do feature extraction, identify and classify the LPI waveform parameters, and distinguish among the various LPI signal modulations. Waveforms used as input signals to the detection algorithm include Frequency Modulated Continuous Wave, Polyphase Codes, Costas Codes and Frequency Shift Keying/Phase Shift Keying waveforms. The output matrices resulting from the most relevant layers of the QMFB tree processing are examined and the LPI modulation parameters are extracted under various signal-to-noise ratios.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

EXECUTIVE SUMMARY

Low Probability of Intercept (LPI) radars are a class of radar systems that have certain performance characteristics that make them nearly undetectable with most common digital intercept receivers. The term low probability of intercept (LPI) is that property of an emitter that because of its low power, wide bandwidth, frequency variability, or other design attributes, makes it difficult to be detected or identified by means of passive intercept devices such as radar warning, electronic support and electronic intelligence receivers. New signal processing techniques are required in order to detect LPI radar waveforms. Consequently, LPI radar systems can operate undetected until the intercept receiver is much closer than the radar's target detector. As a result, the surprise factor remains one of the most important elements of military tactics, which poses a significant problem for the enemy.

To detect LPI radars, new direct digital receivers using sophisticated signal processing are required. This thesis describes a type of spread spectrum intercept receiver and implements a simulation of it. The LPI signal detection receiver is based on Quadrature Mirror Filter Bank (QMFB) Tree processing. The receiver uses orthogonal wavelet techniques and a QMFB tree structure to decompose the input waveform into components representing the energy in rectangular "tiles" in the time-frequency plane. By analyzing the outputs at different layers of the tree it is possible to do feature extraction, identify and classify the LPI waveform parameters, and distinguish between the various LPI signal modulations. Input continuous waveforms used as input signals to the detection algorithm include Frequency Modulated Continuous Wave, Polyphase Codes, Costas Codes and Frequency Shift Keying/Phase Shift Keying waveforms. The output matrices resulting from the most relevant layers of the QMFB tree processing are examined and the LPI modulation parameters are extracted under various signal-to-noise ratios.

The overall architecture of the QMFB presented in this thesis is a good alternative to an intercept receiver, particularly for detecting hopped LPI spread spectrum radar sig-

nals, as the FSK/PSK Costas. The feature extraction gives a time-frequency plane with good estimation of position, center frequency and time duration of the signals processed.

I. INTRODUCTION

A. LPI RADAR SIGNALS AND THEIR DETECTION

In this chapter the LPI radar signals and their detection are explored giving a background of the problem and defining the research objective.

1. To See and Not Be Seen

New advanced components and signal processing algorithms have been developed by modern electronic technology. Some of these new electronic components and signal processing techniques are applied in radar systems with complex waveforms. As a result, the performance of these radars improves significantly. The appearance of modern Electronic Support (ES), Radar Warning Receivers (RWR), and Anti Radiation Missiles (ARM) constitute the most important threats to radar operation on the battlefield. Consequently, modern radars should have a Low Probability of Intercept (LPI) capability in order to hide from and survive an enemy attack.

LPI is considered an important tactical requirement and is being specified by many military personnel using radar today. An LPI radar has specific design features that render it difficult to detect. Low probability of intercept is that property of an emitter that because of its low power, wide bandwidth, frequency variability, or other design attributes, makes it difficult to be detected or identified by means of passive intercept receiver devices. LPI radars attempt to detect targets at longer ranges than intercept receivers at the target can detect the radar. Thus, the objective of an LPI radar is "To See and Not Be Seen," or, "To Detect and Not Be Detected" [1].

2. The Principle of LPI Radar

In order to accomplish interception by an intercept receiver, the detection range of the radar R_R should be longer than the range R_I at which the intercept receiver can detect

the radar emission as shown in Figure 1. From Figure 1, a range factor α can be defined as:

$$\alpha = \frac{R_I}{R_R}. \tag{1.1}$$

If $\alpha > 1$, the radar may be detected at a range greater than the range the radar can detect targets. If $\alpha < 1$, the radar can detect targets at a range greater than the range of intercept. This latter type of radar is an LPI radar.

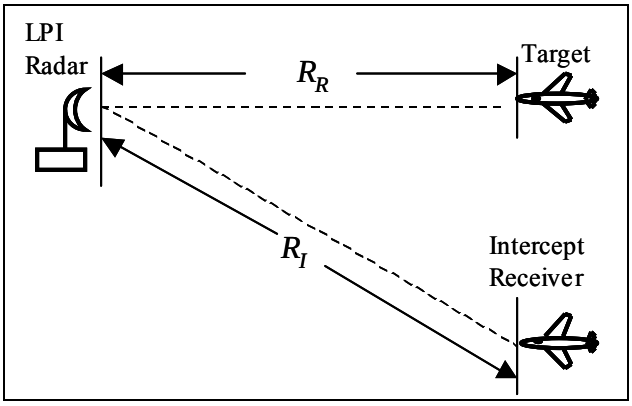

Figure 1 LPI radar, target and intercept receiver configuration. After [2].

It should be noted that R_R and R_I only occur when certain detection probability and false alarm probability conditions are satisfied. In fact, the so-called LPI performance is a probabilistic event. LPI radars may not be detected by certain low sensitivity intercept receivers, but high sensitivity intercept receivers may have no problem. Although, the LPI radars take advantage in the signal-to-noise ratio (SNR) conditions, hiding and spreading the transmitted waveform under the noise level in a wide bandwidth.

In the case of an electronic attack, just detecting the energy will not help program the waveform generator of a jamming system to perform the attack. For this the modulation parameters are necessary and sophisticated signal processing required to defining the transmission waveform parameters of the jamming signal. [2].

3. Characteristics of LPI Radar Signals and Their Detection

LPI radars have many combined features that help prevent their parameter detection by modern intercept receivers. These features are centered on the antenna and transmitter. The first antenna characteristic is a low side-lobe transmit pattern. The use of low side-lobe pattern reduces the possibility of an intercept receiver detecting the radio frequency (RF) emissions throughout the side-lobe structure of the antenna pattern. The second antenna characteristic is the scan pattern, which is precisely controlled to limit the intercept receiver time to short and infrequent intervals (a periodic scan cycle). Scan methodologies can also be added to help confuse identifications of interest if they occur. For example, scan techniques that attempt to confuse identification by an intercept receiver might include amplitude modulation of a monopulse array at conical scan frequencies. [1].

The main drawback of the coherent pulse train waveform is the high ratio of peak to average power emitted by the transmitter. This average power determines the detection characteristics of the radar. For high average power, the short pulse (high resolution) transmitter must have a high peak power, which necessitates vacuum tubes and high voltages. The high peak power transmissions can also easily be detected by ES receivers. In modulated continuous wave (CW) signals, however, the peak-to-average power ratio is one (100% duty cycle) which allows a considerable lower transmit power to maintain the same detection performance. Also, solid state transmitters can be used which are lighter in weight. A comparison is shown in Figure 2.

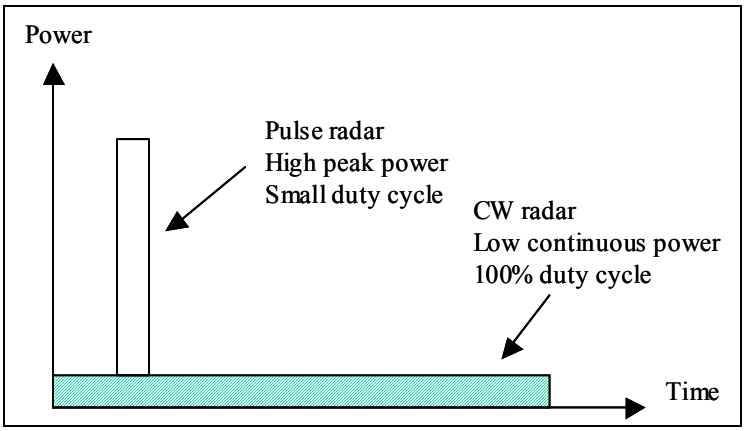

Figure 2 Comparison of pulsed radar and CW Radar. After [1].

LPI emitters use CW signals with large bandwidths and small range resolution cells, resulting in a moderate size target return being spread over multiple cells. Periodically modulated CW signals are extensively used in LPI radar and are ideally suited for pulse compression. They achieve a unity peak-to-average power ratio. The fact that the transmitted signal is continuous does not imply that the portion processed in order to detect a target is infinitely long. There are physical constraints (the illumination time, for example) and processor constraints that can cause this. Fast Fourier Transform (FFT) processors (for frequency-modulated waveforms) and finite duration coherent correlation processors (for phase modulation waveforms) are among the most often used as well as combinations of both [1].

There are many wideband modulation techniques available from the transmitter function that provides an LPI transmit waveform. Any change in the radar's signature (pulse repetition frequency (PRF), pulse width (PW), carrier frequency, polarization, scan modulation) can help confuse an intercept receiver and make identification difficult. Wideband modulation techniques include:

- Linear and Non-Linear frequency modulation,
- Costas arrays, frequency hopping,
- Phase modulation (polyphase coding),
- Combined phase shift keying, frequency shift keying (PSK, FSK), and
- Pseudo-noise modulation.

The radiated energy is spread over a wide frequency range in a noise-like manner with most of these modulation techniques.

Another feature of an LPI transmitter is power management. This is one of the benefits to using solid-state radars. The ability to control the signature emitted by the array is used to limit the emissions to the appropriate Range/Radar Cross Section requirement. The emissions are also limited in time (short dwell time). With the use of wideband CW emissions it is only necessary to transmit a few watts instead of tens of kilowatts of peak power required by pulsed radars with similar performance. It is important to note that the radar's ability to detect targets depends not on the waveform characteristics but

on the transmitted energy density returned to the radar from the target. The main objective of a LPI radar is to operate under low (SNR) conditions so that integration of the signal over several contiguous range cells can be used to detect and track the targets of interest. This integration leads to an operative resolution cell that is larger than FFT bin [1].

Another approach to achieving a lower probability of interception is to interleave the LPI radar mode with an infrared sensor (dual mode approach). The amount of time that the RF transmitter is radiating is therefore reduced.

4. Research Objective

This thesis developed a simulation of a method to decompose a signal, using orthogonal basis functions and a quadrature mirror filter bank (QMFB) tree configuration. Detailed information about processed LPI signals were then extracted. The receiver for this approach is shown in Figure 3. In this block diagram a received waveform is bandpass filtered and sampled at the Nyquist rate. The digital sequence is then fed to the QMFB tree where it is decomposed and matrices of weights are output from each layer. These weights are then squared to produce coefficients representing the energy in each portion of the waveform.

The information is then analyzed to determine all the relevant parameters from the waveform. Next, the output of the analyzer is a list of all parameters already extracted. The classifier takes the list from the analyzer, determines which parameters belong to common transmitters, and outputs a list of possible transmitters, their types, and parameters. Further filtering could be done at this point, eliminating false alarms and signals that are not of interest to the interceptor. The classifier may even be adaptive and change classification criteria based on current and previous input and results [4].

The research in this thesis was primarily concerned with the receiver's QMFB tree and the analyzer block. It was also focused on developing algorithms to detect and determine the features for each type of LPI signals. The performance of the QMFB under various SNR is also investigated in the presence of additive White Gaussian Noise

(WGN). The only signals assumed to be present are single LPI signals at a time. The input bandpass filter and sampler in the receiver shown in Figure 3 are assumed to be ideal.

This thesis does not examine the classifier function in any detail. The primary job of the classifier, however, is to take the list of parameters, determine the most likely number of transmitters, and group the parameters to the transmitters. This specific task was not part of this thesis.

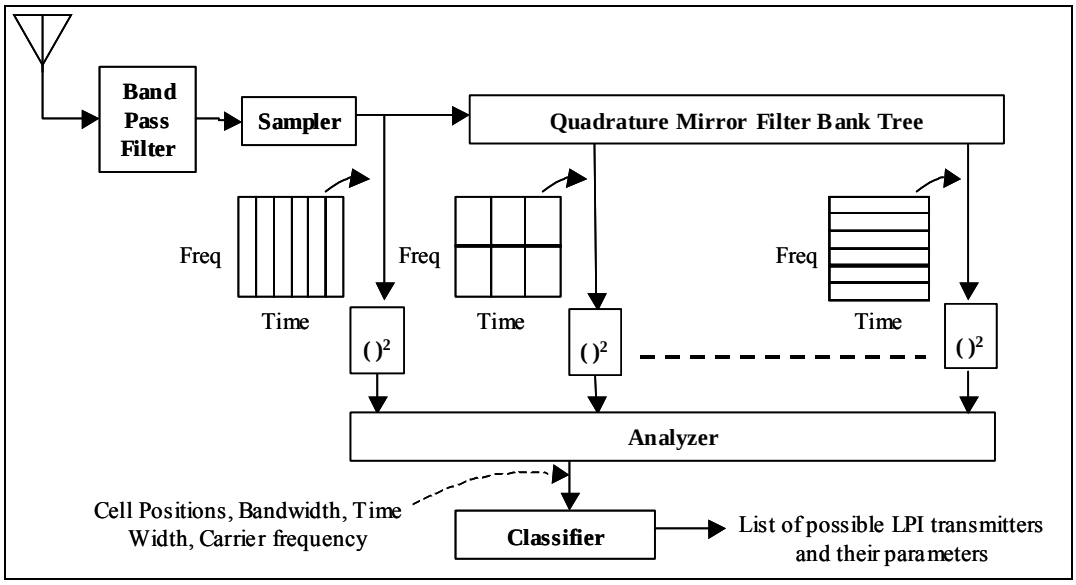

Figure 3 LPI receiver block diagram. After [4].

B. PRINCIPAL CONTRIBUTIONS

The first phase of this thesis involved doing a literature research to reach a good understanding on LPI radar signals, quadrature mirror filter (QMF), signal processing and MATLAB® coding [5]. This was found through different books, papers and courses from databases as IEEE, some signal processing books, and courses taken during the studies in Electrical and Computer Department of NPS, respectively. The next phase was to define the QMFB signal processing and develop a new program to examine LPI radar signals. With the new program in MATLAB®, test signals were used to verify that the signal

processing worked correctly. Next, a list of different LPI signals were analyzed with the program, which displays time-frequency output plots from which it is possible to extract the different parameters of the LPI signals. A comparison of the results of the processing done by the QMFB tree on all the signals is presented as a QMFB processing performance on LPI radar signals. Finally, a summary of the thesis, conclusions of the QMFB processing and recommendations to continue with this research are indicated.

C. THESIS OUTLINE

Chapter II gives a brief overview on the theory and background on time-frequency signal processing and wavelets decomposition. Chapter III presents the LPI signals processed by the QMFB tree where each signal is analyzed and the parameters are extracted from the most relevant QMFB output layer finishing with an analysis and comparison of the signal processing results. Finally, Chapter IV gives conclusions from the QMFB processing, and recommendations for future research are indicated.

THIS PAGE INTENTIONALLY LEFT BLANK

II. QUADRATURE MIRROR FILTER BANK

This thesis investigates an LPI intercept receiver based on a linear decomposition of the received waveform through a Quadrature Mirror Filter Bank (QMFB) Tree using wavelet filters. The receiver provides good detection of the LPI signal parameters in an effort to distinguish between the different modulations. This chapter presents the QMFB theory, discussing the mathematical background for signal detection and the waveform decomposition using Wavelets, and finishes with a description of the MATLAB® codes used to implement the QMFB tree.

A. BACKGROUND

1. Time-frequency Decomposition

Various methods of decomposing a waveform on the time-frequency plane have recently been investigated. The most common methods can be divided into linear and bilinear transforms. The Short Time Fourier and Wavelet Transform are examples of linear transforms, and the Wigner transform is example of a bilinear transform. Wigner transforms are called bilinear because the input waveform appears twice in the development of the transform. Better resolution occurs in the time-frequency plane than with linear techniques. However, the computational burden is greatly increased resulting in other side effects. At the same time, is not easy to find studies specifically using this transform for the detection of LPI waveforms. With the computational complexity and the possibility of confusing cross-terms, the bilinear transform is considered further in [18]. Linear transforms of the continuous time signal $f(t)$ have the following form

$$a_k = \int f(t)\Phi_k(t)\, dt \tag{2.1}$$

where $\Phi_k(t)$ is the basis set, t is the time index, and k is the function index

The Fourier transform, for example, has a basis set consisting of sines and cosines of frequency $2\pi k$. The basis functions are said to be orthonormal if,

$$\int \Phi(t-k)\Phi(t-l)dt = \begin{cases} 1 & k = l \\ 0 & k \neq l. \end{cases} \quad [6] \tag{2.2}$$

If the basis functions are orthonormal, there is no redundancy in the representation of the signal $f(t)$. It is possible to sample the input waveform at the Nyquist rate and retain all the information. In that case, the time variable, t, in (2.1) and (2.2) should be considered to be discrete and the integral should be replaced with summations. The basis functions are said to be orthogonal if

$$\int \Phi(t)\Phi(t-k)dt = E\delta(k) = \begin{cases} E & \text{if } k = 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

where E stands for energy of $\Phi(t)$ [7].

Wavelet basis functions are effectively non-zero for only a finite time interval, and can be designed to satisfy (2.2). References [4] and [6] demonstrate that these orthogonal Wavelets can be implemented using Quadrature Mirror Filters: filter pairs that are designed to divide the input signal energy into two orthogonal components based on the frequency. The basis function becomes a contracted wavelet, or a short high frequency function as it is in Figure 4 (a) where , and the wavelet transform divides the time-frequency plane into tiles as shown in Figure 4 (b). Here the area of each tile represents (approximately) the energy within the function (rectangular regions of the frequency plane). Note that not all of the signal's energy can be located in a single tile because it is impossible to concentrate the function's energy simultaneously in frequency and time. A characteristic of the Wavelet transform is that the tiles become shorter in time and occupy a larger frequency band as the frequency is increased.

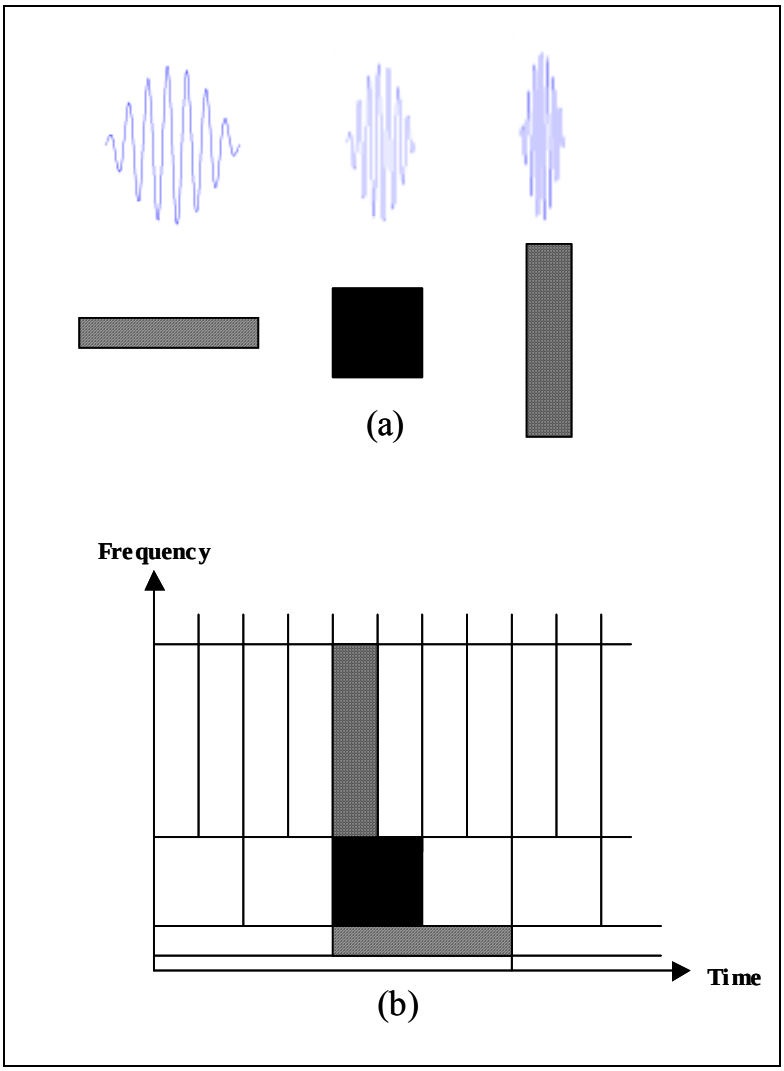

Figure 4 Basis functions and time-frequency resolution of the wavelet transform. (a) Basis functions. (b) Coverage of time-frequency plane.

Using Wavelet techniques to develop an appropriate basis set and a QMFB for implementation, it is possible to decompose the waveform in such a way that the tiles have the same dimensions regardless of the frequency [4, 10]. Since the transform is linear, there is a fundamental limit on the minimum area of these tiles. However, the nature of the QMFB configuration is such that each layer outputs a matrix of coefficients for tiles that are twice as long (in time) and half as tall (in frequency) as the tile in the previous layer. By properly comparing these matrices, it is possible to extract signal features using both fine frequency and fine time resolutions. Parameters such as bandwidth, center frequency, energy distribution, phase modulation, signal duration and location in the time-frequency

plane can be determined using these techniques, making them valuable for intercept receivers in order that they can determine the number of transmitters present and which types are in operation.

2. Decomposition of Waveforms

a. Short Time Fourier Transform

Complex sinusoids are used by the Fourier transform to perform the analysis of signals using appropriate basis functions. This approach is difficult due to the infinite extent of the basis functions as any time-local information, such as an abrupt change in the signal, is spread out over the entire frequency axis. This problem has been addressed by introducing windowed complex sinusoids as basis functions [10]. This leads to the doubly indexed windowed Fourier transform:

$$X_{WF}(\omega,\tau) = \int_{-\infty}^{\infty} e^{-j\omega t} w(t-\tau) x(t)\, dt \tag{2.4}$$

where $w(t-\tau)$ constitutes an appropriate window. $X_{WF}(\omega,\tau)$ is the Fourier transform of $x(t)$ windowed with $w(\cdot)$ shifted by τ. The major advantage of the windowed or short-time Fourier transform (STFT) is that if a signal has most of its energy in a given time interval $[-T,T]$ and frequency interval $[-\Omega,\Omega]$, then its STFT will be localized in the region $[-T,T]\times[-\Omega,\Omega]$ and will be close to zero in time and frequency intervals where the signal has little energy [10]. A limitation of the STFT is that, because a single window is used for all frequencies, the resolution of the analysis is the same at all locations in the time-frequency plane. The possibility of having arbitrarily high resolution in both time and frequency is thus excluded.

b. Wavelet Transform

By varying the window used, resolution in time can be traded for resolution in frequency. To isolate discontinuities in signals it is possible to use some basis

functions, which are very short, while longer ones are required to obtain a fine frequency analysis. One method to achieve this is to have short high-frequency basis functions, and long low-frequency basis functions [10]. This is exactly what the wavelet transform achieves where the basis functions are obtained from a single prototype wavelet by translation and dilation/contraction:

$$h_{a.b}(t) = \frac{1}{\sqrt{a}} h\left(\frac{t-b}{a}\right) \tag{2.5}$$

where a is a positive real number and b is a real number. For large a, the basis function becomes a stretched version of the prototype wavelet (low frequency function); while for small a, the basis function becomes a contracted wavelet (short high frequency function) as it is in Figure 4 (a). The wavelet transform (WT) is defined as

$$X_W(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} h^*\left(\frac{t-b}{a}\right) x(t)\, dt \tag{2.6}$$

The time-frequency resolution of the WT involves a different tradeoff than that used by the STFT. At high frequencies, the WT is sharper in time; while at low frequencies, the WT is sharper in frequency [10]. If the parameters (ω, τ) and (a,b) are continuous, the STFT from Equation (2.4), and the WT, from Equation (2.6), are highly redundant. Therefore, the transforms are usually evaluated on a discrete grid on the time-frequency and time-scale planes, respectively, corresponding to a discrete set of continuous basis functions. At this point a grid is needed such that the set of basis functions constitutes an orthonormal basis (no redundancy). Unfortunately, for the STFT, this only occurs if $w(t-\tau)$ is badly localized in either time or frequency. This is the reason that the STFT is usually oversampled, as a redundant set of points is used, so that better behaved window functions can be used. In the wavelet transform case, however, it is possible to design "practical" functions $h(\cdot)$ such that the set of translated and scaled versions of $h(\cdot)$ forms an orthonormal basis. "Practical" means that the function should be at least continuous, and perhaps with continuous derivatives also. Making the translation and dilation/contraction discrete parameters of the wavelets in Equation (2.5):

$$h_{mn}(t) = a_0^{-m/2} \cdot h(a_0^{-m} t - n b_0) \tag{2.7}$$

where m, n are integers, $a > 1$, and $b_0 \neq 0$, which correspond to $a = a_0^m$ and $b = na_0^m b_0$. Note that the translation step depends on the dilation, since long wavelets are advanced by large steps, and short ones, by small steps. On this discrete grid, the wavelet transform is thus

$$X_W(m,n) = a_0^{-m/2} \int_{-\infty}^{\infty} h(a_0^{-m}t - nb_0)x(t)\, dt \tag{2.8}$$

Of particular interest is the discretization on a dyadic grid, which occurs for $a_0 = 2$, $b_0 = 1$. It is possible to construct functions $h(\cdot)$ so that the set

$$h_{mn}(t) = a_0^{-m/2} \cdot h(a_0^{-m}t - nb_0), \tag{2.9}$$

where m, n are integers, $a_0 = 2$, and $b_0 = 1$ is orthonormal [10].

A classic example is the Haar basis, which is not continuous, but is of interest because of its simplicity, where:

$$h(t) = \begin{cases} 1 & 0 \le t < 1/2 \\ -1 & 1/2 \le t < 1 \\ 0 & \text{otherwise.} \end{cases} \tag{2.10}$$

The orthonormality is easily verified since at a given scale, translates are nonoverlapping, and because of the scale change by 2, the basis functions are orthonormal across scale. The Haar basis is shown in Figure 5. However, the Haar function is discontinuous and is not generally appropriate for signal processing. A continuous set of basis functions must be used [6].

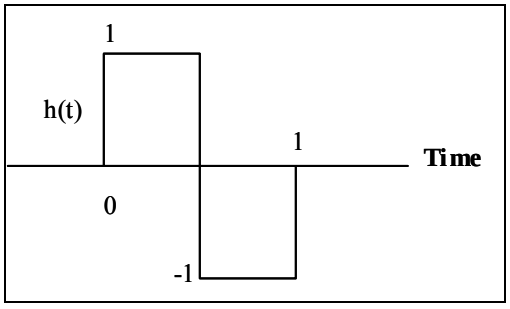

Figure 5 Mother wavelet for the Haar basis set.

From a signal processing point of view, a wavelet is a bandpass filter. Therefore the wavelet transform can be interpreted as a constant-Q filtering with a set of subband filters, followed by a sampling at the respective Nyquist frequencies corresponding to the bandwidth of the particular subband [6].

3. Discrete Two-Channel Quadrature Mirror Filter Bank

In many applications, a discrete-time signal $x[n]$ is first split into a number of subband signals $\{v_k[n]\}$ by means of an analysis filter bank. The subband signals are then processed and finally combined by a synthesis filter bank, resulting in an output signal $y[n]$. If the subband signals are bandlimited to frequency ranges much smaller than that of the original input signal, they can be down-sampled before processing. Due to the lower sampling rate, the processing of the down-sampled signals can be carried out more efficiently. After processing, these signals are up-sampled before being combined by the synthesis bank into a higher-rate signal. The combined structure employed is called a QMFB. If the down-sampling and the up-sampling factors are equal to or greater than the number of bands of the filter bank, then the output $y[n]$ can be made to retain some or all of the characteristics of the input $x[n]$ by properly choosing the filters in the structure [11].

a. The Filter Bank Structure

Figure 6 shows the basic two-channel QMFB. Here, the input signal $x[n]$ is first passed through a two-band analysis filter bank containing the filters $H_0(z)$ and $H_1(z)$, which typically have lowpass and highpass frequency responses, respectively, with a cutoff frequency at $\pi/2$ as indicated in [11].

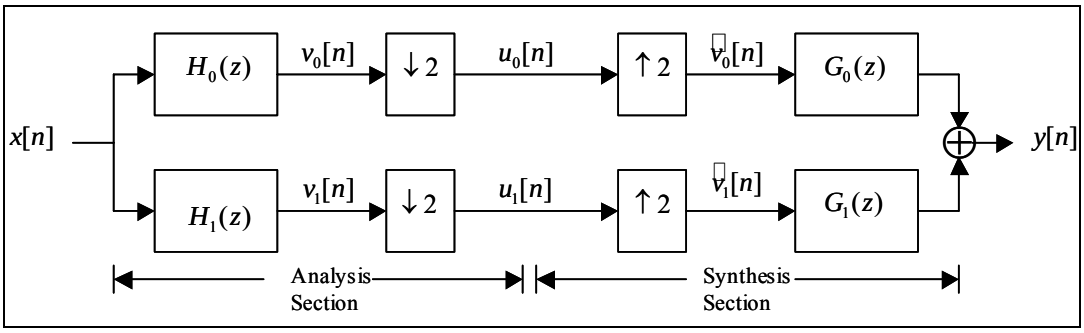

Figure 6 The two-channel quadrature mirror filter (QMF) bank. After [11].

The subband signals $\{v_k[n]\}$ are then down-sampled by a factor of 2 in the "signal analysis section" to be transmitted to the "signal synthesis section" where the signals will be up-sampled by a factor of 2 and passed through a two-band synthesis filter bank composed of the filters $G_0(z)$ and $G_1(z)$ whose outputs are then added yielding $y[n]$. It follows from the figure that the sampling rates of the input signal $x[n]$ and output signal $y[n]$ are the same. The analysis and the synthesis filters in the QMFB are chosen so as to ensure that the reconstructed output $y[n]$ is a reasonable replica of the input $x[n]$. Moreover, they are also designed to provide good frequency selectivity to ensure that the sum of the power of the sub-band signals is reasonably close to the input signal power [11]. For the signal analysis section, $H_0(\omega)$ is a lowpass filter and $H_1(\omega)$ is a mirror-image highpass filter as is shown by Figure 7.

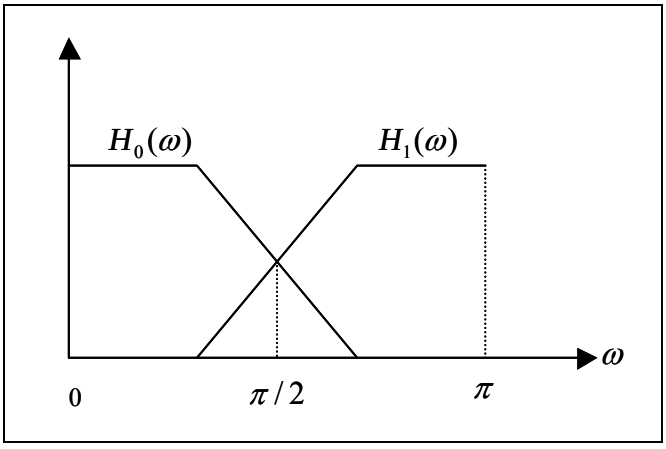

Figure 7 Typical frequency response of the analysis filters. After [11].

4. Filtering and Arbitrary Tiling

a. Filtering the Lowpass Component

Finite impulse response (FIR) filters and decimators can be arranged in the tree structure as shown in Figure 8 to effect an orthogonal wavelet decomposition of a signal [12]. The discrete input waveform is denoted as the sequence $\{c_0\}$ and the output sequences of each branch are as shown in the figure. Since each branch of the tree down samples by 2, each sequence will have half as many elements as the preceding sequence.

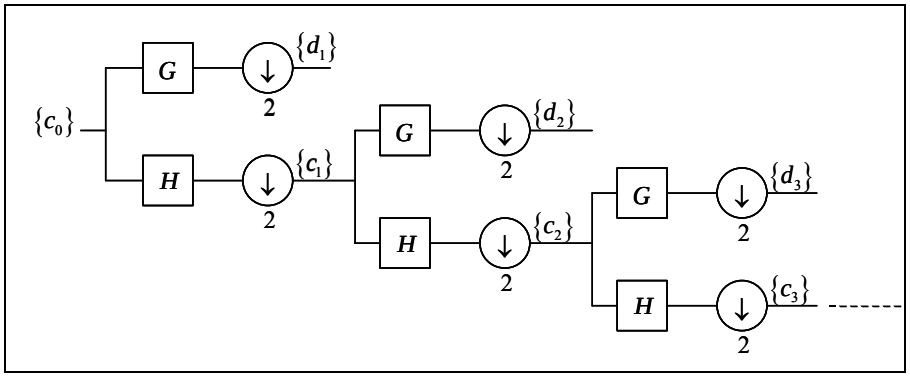

Figure 8 Wavelet filter bank tree [12].

A filter tree using the same orthogonal pair of filters throughout, and with equal length branches, as in Figure 8, yields a rectangular tiling diagram. The time-frequency tiling diagram shown in Figure 9, is one method that can be used to describe this decomposition. Time-frequency tile is "the region in the plane, which contains most of that function's energy" [4]. However, not all of a function's energy can be located in a tile because it is impossible to fully concentrate energy simultaneously in time and frequency.

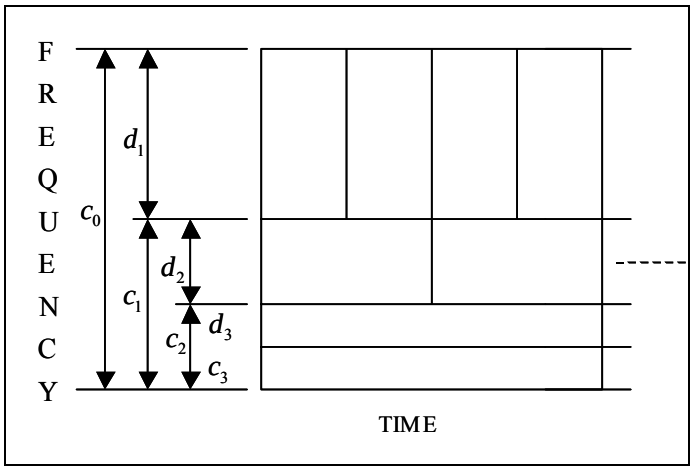

Figure 9 Time-frequency diagram for the wavelet filter bank tree [12].

b. Filtering the Highpass Component (Arbitrary Tiling)

The last section demonstrated that by cascading filters and filtering the lowpass component of the previous output, a tiling with finer frequency resolution at lower frequencies was achieved. In many cases concerning detection, however, this is not desirable. Many man-made signals, for example, will have a constant bandwidth over a wide range of center frequencies. Instead, an arbitrary tiling is desired to meet specific requirements for the type of signal to be detected. In order to accomplish this, it is possible to modify the filter bank presented in the last section.

Consider the cascading filter diagram in Figure 10 where, instead of filtering the lowpass output of each stage, the highpass output is filtered. Again, the input sequence is split at each stage into high-frequency and low-frequency orthogonal sequences, and the tiling diagram shown in Figure 11 is therefore obtained and is sometimes referred to as “Wavelet Packet Tiling” [4].

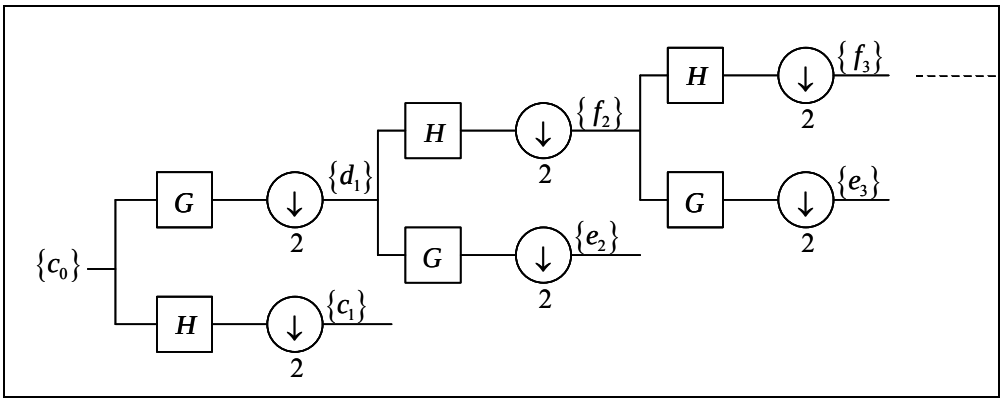

Figure 10 Wavelet filter bank tree.

Notice the second and third layer seem to be flipped in Figure 10. The figure is drawn so that the output sequence at the top of the drawing contains the highest frequency components of the input sequence. This will be important later. To understand why they are flipped, consider the aliased frequency spectrum of the filters, shown in Figure 12. The output from the G filter in the first layer contains the higher frequency components of the original sequence, but shifted, so it is actually the DC component of the output of G. The result is that the output of G is frequency reversed, much like the lower sideband of a single sideband communications system. Of course, similarly further down the cascade will "unflip" the signal. A simple rule exists that maintains the integrity of the output. This will be discussed shortly.

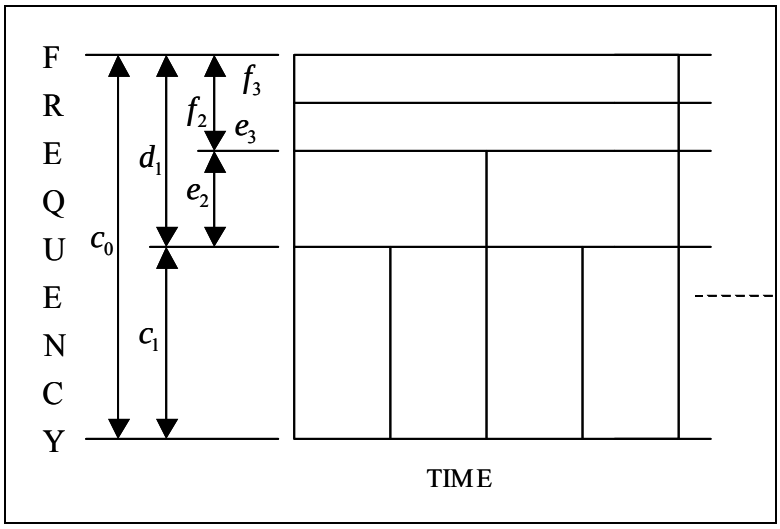

Figure 11 Time-frequency diagram for the wavelet filter bank.

It is possible to create another tiling scheme by combining the Wavelet filter bank and Wavelet tiling, as demonstrated in Figure 13. The construction rule for this figure, in order to keep the higher frequency outputs of each branch above the lower frequency outputs, is to count the number of *G* filters up to the branch. If the number is even, the next *G* filter will output the high frequencies. If odd, the next *H* filter will output the high frequencies.

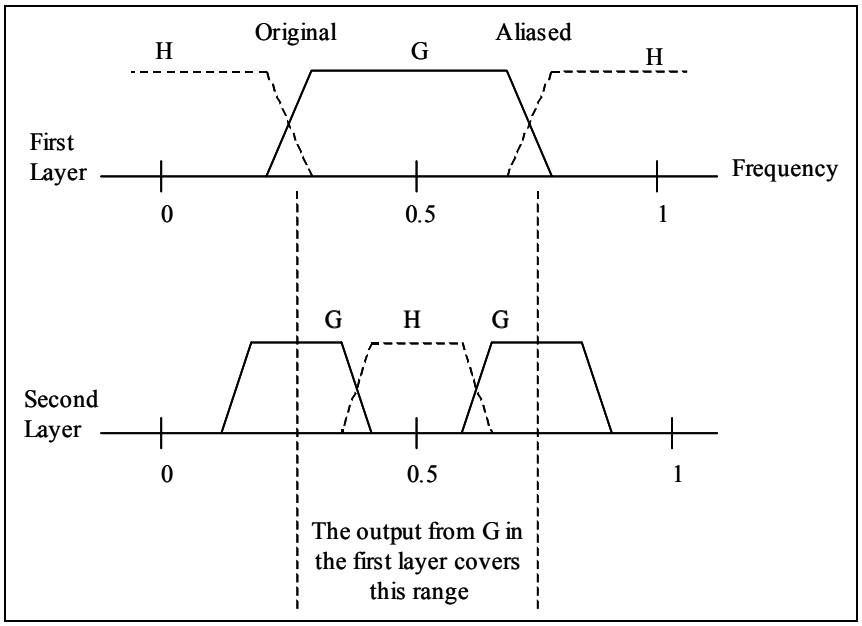

Figure 12 Response of filters.

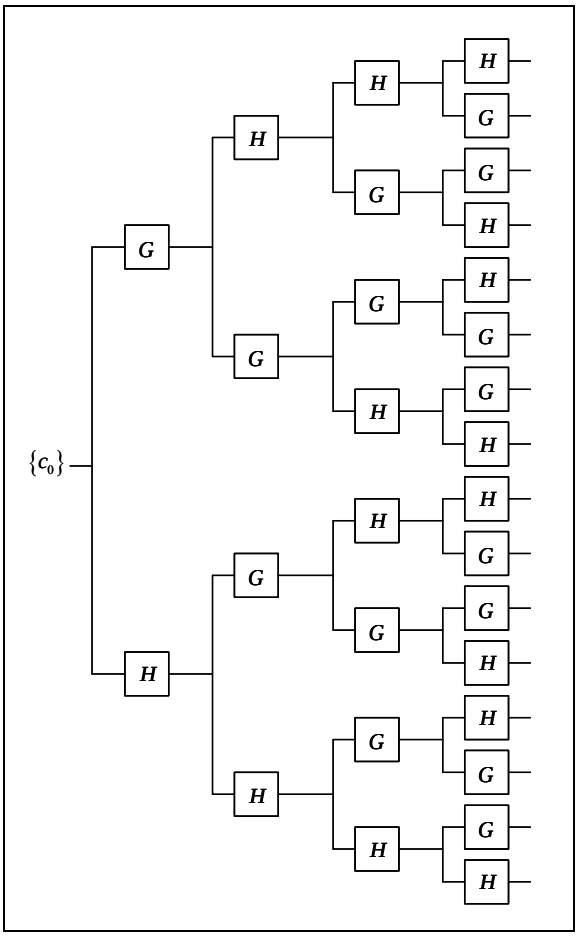

Figure 13 Combining the wavelet filter bank and wavelet tiling. (Decimation by 2 is included in each filter box).

5. Wavelet Filters

The objective of this section is to find a FIR wavelet filter that best approximates the perfect time-frequency tiling by trading off and minimizing the out-of time and out-of frequency energy from a comparison in between the Haar filter and the Sinc filter.

a. The Haar Filter

The Haar filter was discussed briefly at the end of Section 2.a. It has two coefficients

$$h(0) = h(1) = 1/\sqrt{2} \tag{2.11}$$

The energy in each output value is equal to the low-pass energy from the two corresponding input values and from no others. Since each of the two input values contributes equally to the output, the pass region is also flat along the time dimension. It is the only Wavelet FIR filter that is symmetric. The filter perfectly concentrates (tiles) the input energy in time. However, the Haar filter does not tile well in frequency [4, 15].

b. The Sinc Filter

The Sinc filter perfectly concentrates energy in frequency. It has however, an infinite number of coefficients. This condition is modified in this thesis. By starting with the desired frequency response, which is a flat passband, an infinitely narrow transition, and a zero across the stop band, the inverse Fourier Transform is taken. This will result in a sinc function in the time domain [4, 15]. Therefore, the sinc function is

$$\mathrm{sinc}(k) = \begin{cases} \dfrac{\sin(\pi k)}{\pi k} & k \neq 0 \\ 1 & k = 0 \end{cases} \tag{2.12}$$

Since the passband ranges from $-\pi/2 < \omega < \pi/2$ or $-0.25 < f < 0.25$, the nulls of the sinc function will be at $2T$ for a sampling period of T to obtain the filter coefficients. The sinc function will be sampled at the normalized sampling rate of $T = 1$ for a situation similar to that shown in Figure 14. One way to sample the function would be to let the main tap sample occur at the center of the main lobe. However, two main taps are needed and their sum needs to be as large as possible. This occurs for the sinc function if both main tap samples are equally spaced about the center of the main lobe. The sum of the square of the coefficients must be unity also, which is achieved by scaling the sinc by $1/\sqrt{2}$, giving

$$h(n) = \frac{1}{\sqrt{2}} \mathrm{sinc}\left(\frac{n + 0.5}{2} \right) \tag{2.13}$$

where n is an integer.

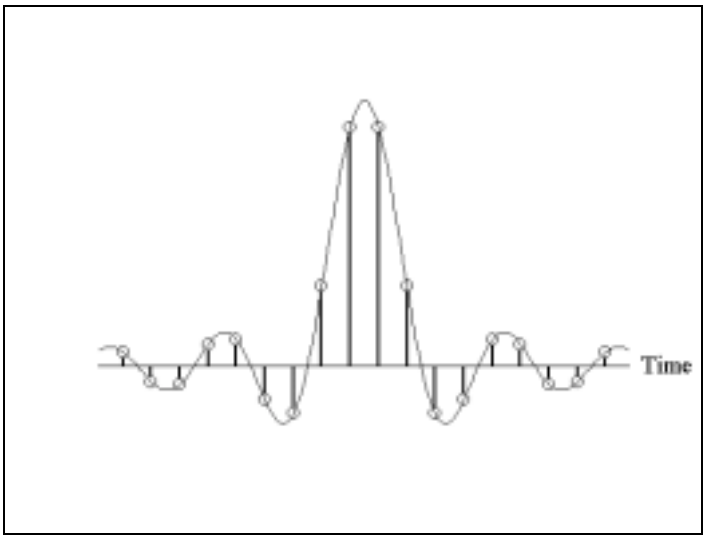

Figure 14 Sampling under a sinc envelope.

In fact, this filter meets the criteria of Wavelet filters. The only problem is that there are an infinite number of coefficients. A small amount of non-orthogonality will occur when trying to determine how to truncate this filter and maintain a good frequency response. Some cross-correlation will take place between filters.

If the ends of the filter are simply truncated, some ripples in the passband of the frequency response will appear which is sometimes called the "Gibb's Phenomena". It is a well-known result of this type of truncation [12]. Multiplying the coefficients by a rectangular shaped window in the time domain can be viewed as a convolution of the perfect filter response with a sinc function (the Fourier Transform of the rectangular window) in the frequency domain.

The solution is to use a non-rectangular window and one whose Fourier Transform has a narrower main lobe and smaller sidelobes than the sinc function. The Hamming window is one that is commonly used [11]. Multiplying the coefficients from (2.13) by this window, and using the results in a FIR filter, the frequency response needed is generated. Energy will be lost at the filter transitions, which is primarily the result of the loss of orthogonality from truncating the filter. For detection, instead of losing the energy at those frequencies, a better trade-off would be a small amount of cross-correlation between the filters so that some energy appears in more than one tile.

To achieve this type of prototype filter the impulse response can be modified to have a passband that is slightly greater than $\pi/2$. Thus, the *H* and *G* filters are squeezed together slightly. This can be achieved by compressing the sinc envelope of (2.13) slightly. At the same time, it will be desirable to rescale the coefficients slightly, so the sum of the squares equals one. These modifications to (2.13) give

$$h(n) = \sqrt{\frac{S}{2}}\text{sinc}\left(\frac{\text{n+0.5}}{\text{C}}\right)w(n) \tag{2.14}$$

where $-N/2 \le n \le (N-2)/2$, *C* is the compression variable, *S* is the scaling variable, *N* is the number of coefficients, and $w(n)$ is the Hamming window to suppress the Gibb's phenomena. An infinite number of coefficients and no window could be used. *C*=2 and *S*=1 would create orthogonal wavelets filters. For these filters, the greatest cross correlation occurs between tiles in the same frequency band and adjacent in time when *N*=512 (the number of coefficients), with values *C*=1.99375872328059, *S*=1.00618488680080, and a Hamming window giving a cross correlation of less than 0.001. This is called the "modified sinc filter" [12], [14].

B. QUADRATURE MIRROR FILTER BANK (QMFB) TREE

This section describes how the QMFB tree is implemented using the theory in the previous section.

1. The Receiver

Orthogonal wavelet decomposition of the unknown signal can be implemented using QMFs by designing filter pairs to divide the input signal energy into two orthogonal components based on frequency. The tiles are used to refer to the rectangular region of the time-frequency plane containing the basis function's energy. By arranging the QMF pairs in a fully developed tree structure it is possible to decompose the waveform in such a way that the tiles have the same dimensions within each layer [4, 15]. Thus, every filter output is connected to a QMF pair in the next layer as shown in Figure 15.

The architecture of the QMFB tree used in this thesis is illustrated in Figure 15. Each QMF pair divides a digital input waveform into its high frequency and low frequency components with a transition centered at $\pi/2$. A normalized input of one sample/second is assumed, with a signal bandwidth of $[0,\pi]$. Since each filter output signal has half the bandwidth, only half the samples are required to meet the Nyquist criteria, therefore these sequences are then down sampled by two. The same number of output samples, as were input are returned. For example if 100 samples appear at the input of the first QMF pair, 100 samples appear at the output. Each of the two resulting sequences are then fed into QMF pairs forming the next layer where the process is repeated, and so on down the tree.

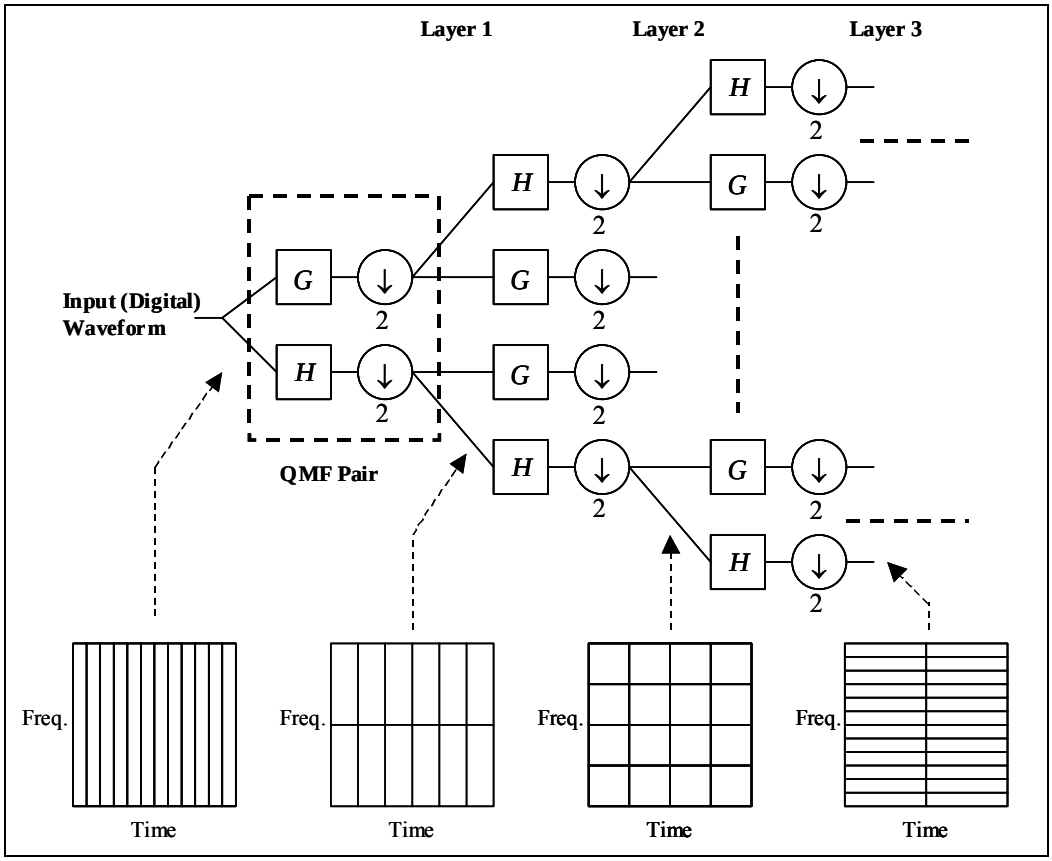

Figure 15 Quadrature mirror filter bank (QMFB) Tree. After [4].

Considering the square of each element of the input waveform represents the waveform's energy for that sample, each represents the energy contained in the corresponding tile in the left most time-frequency diagram shown in Figure 15. Similarly, the outputs from each layer of the tree form a matrix whose elements, when squared ap-

proximately represent the energy contained in the tiles of the corresponding time-frequency diagrams shown in the figure. As shown in Section A.5, it is not possible to find a filter that perfectly divides the energy into tiles in both time and frequency [4, 13].

When the waveform consists entirely of WGN, the tile's energy will have random values with a Chi-squared probability distribution. When a deterministic signal is added, tiles containing energy from the signal will have probability distribution that is Chi-squared with non-centrality parameters and will, therefore, tend to have larger mean values and thus make threshold detection a possibility [4, 13].

Since the transform is linear, a fundamental limit on the minimum area of each of the tiles exists. However, looking at Figure 15, it can be noted that each layer outputs a matrix of energy values for tiles that are twice as long (in time) and half as tall (in frequency) as the tile in the previous layer. By properly comparing these matrices, it is possible to find concentrations of energy and estimate their position and sizes with high resolution in both time and frequency. Using these techniques, then, a waveform can be decomposed and the bandwidths, the time widths, and locations in the time-frequency plane can be estimated. All this information, of course, can then be used by the interceptor to decide how many transmitters, and which types, are in operation [4, 13].

The receiver block diagram is shown in Figure 3 (see p. 6). A received waveform is bandpass filtered and sampled at the Nyquist rate. The digital sequence is then fed to the QMFB tree where it is decomposed. Matrices of values are output from each layer, and are then squared to produce numbers representing the energy in each tile.

2. Filters

When considering only FIR filters, some requirements are necessary. First, the filters must meet the wavelet requirements described in Section 2. Basically, these restrict the possible filter coefficient values to ensure:

1) An orthogonal decomposition so that the energy in sequences output from each QMF pair will equal the energy input.

2) That the output from the *H* filter, as labeled in Figure 15, consists of low frequency components of the input, while the output from the *G* filter consists of high frequency components.

As it turns out, one practical consequence of these requirements is that, when a suitable H filter is found, the *G* filter is obtained by negating and time reversing every other coefficient value.

Second, the filters approximately collect energy in tiles. Essentially, they must pass as much energy from inside a tile as possible, while rejecting as much as possible from outside a tile with a reasonably flat pass region.

Some filters, such as the Haar filter, meet the wavelet requirements that perfectly tile the input energy in time, but unfortunately, does not tile well in frequency. The opposite of the Haar filter, in this respect, would be the sinc filter. Both were described in Section A.5. The correct filter is the "modified sinc filter", which will return a good tile in time and frequency [4, 13].

C. SIMULATION PROGRAMS

Some of the programs used in the simulations for this thesis are described in this section. The simulations are used to verify mathematically derived results and to provide results when the mathematics becomes extremely difficult. All these simulations were done on a Pentium 4-based personal computer, with a 2.0 GHz CPU and 1 Gb of RAM using MATLAB® 6.1 for Windows 2000.

The organization of the simulation programs follows the receiver block diagram in Figure 3. The waveform is assumed to be properly bandpass filtered and sampled at the Nyquist rate. The MATLAB® code for generating the sampled input wavefoms (BPSK, Frank Code, Polyphase, FMCW, and Costas signals) can be found in Reference [17]. The input signals are two-row data containing the first one called *I,* which is the real part of the signal, and the second is called *Q* and is the imaginary part of the signal generated. The signals generated are used as input into the QMFB program, which is described next

in this section and carries out the function of the QMFB Tree. The output is then written to the hard drive. The QMFB tree output data, from the hard drive, can then be analyzed using code written to execute the feature extraction from every processed waveform.

1. Quadrature Mirror Fiter Bank Tree Programs and Procedures

Appendix A contains the m file *startpoint.m* that loads the signal, which will be filtered by the QMFB (all programs and program variables are shown in bold). The code requires the sampling frequency *fs* and, in case the signal length does not correspond to a power of two in the number of samples, the signal will be padded with zeros until the length reaches the next power of 2. In this manner the amount of layers resulting from the processing has direct relationship with the length of the signal (number of samples) in the sense that the number of layers is determined by the power of 2 that the length of the signal has. For example, if a signal row data has a length of 2800 samples, this signal is padded with zeros until reach the next power of 2, which is $2^{12} = 4096$ samples, indicating that the number of layers resulting from the QMFB processing is 12 layers (*lay*=12). Afterwards, the code filters the signal applying the *qmfb.m* function.

Appendix A also contains the m file function *qmfb.m* that decomposes the input sequence as described in Section A (Background). The function takes the incoming signal *f*, formally the signal *tt* from the *startpoint.m*, and processes it with a QMFB tree structure, where *filter* specifies the file containing the filter coefficients of the modified sinc filter to be applied through the QMFB tree over the signal. Output sequences from each layer of the QMFB are written to files in the same directory where the function resides on the hard drive where variable *lay* is replaced by the layer number. These are ASCII files and the data is stored in a two-matrix format. The first matrix is called *R* and contains the real part of the data and the second matrix is called *Q* and contains the imaginary part of the data in each column representing the output from a particular filter in the layer. Frequency is represented across each line in the file (lowest frequency to the left) and time is represented down the file (lower values representing later time).

The number of layers is then determined from the length of the input sequence. The input matrix of each layer *I* is initially set equal to the input sequence and the output

matrix for each layer, out, is set equal to *I* with the intention to save memory. The code then loops once for each layer of the QMFB. Inside the loop, the layer number is first displayed on the screen to indicate to the user what the program is doing and *flag* is initialized to 1. Then, *out* is reshaped to match the dimensions required for the output of the layer.

The code then loops again inside the first loop once for each column of the current input matrix. *G* and *H* are the output sequences from each filter pair. As described in Sections 3 and 4, these sequences represent a decomposition of the input into lower and upper frequency bands, which alternates down the layer. This is tracked by a flag and the sequences are written to *out* in the correct order. At the end, the output for the current layer, which is a matrix, is used as the input for the next, and the output data is written to disk.

Appendix A additionally contains the m file *tsinc.m* that uses *N* coefficients generated by the m file *tsinc_su.m* also listed in Appendix A. This file generates the coefficients described in Equation (2.14) under the *h.dat* file (low pass filter) and *g.dat* file (high pass filter) for a modified sinc filter with a Hamming window. These are ASCII files and the data is stored in double precision format as a matrix. Once generated, the coefficients are saved to the hard drive and called upon as needed by *tsinc.m*, which uses them to decompose the column vector *c0* into a column vector *d1* (high frequency) and into a column vector *c1* (low frequency) using a built-in MATLAB® function called *filter* which is designed to work like a FIR filter and **i** is used to accomplish the decimation by 2.

The *startpoint.m* file, contained in Appendix A, automatically displays the time-frequency plane from 4 particular layers through the QMFB processing. The first figure displays 4 contour plots of the selected layers. The second figure displays the same 4 layers using the pcolor MATLAB® command (4 contour plot in a figure and 4 color plots in other). The third figure displays the same 4 layers in a 3 dimensions view using the mesh MATLAB® command. The fourth and last figure displays the 4 layers in a 3 dimensional view using the surf MATLAB® command. That gives a total of four figures each with four plots.

In the case of all output matrix layers need to be plotted, four more files are provided in Appendix A. The *multiple_colorplot.m* file displays all time-frequency plane from all the layers using the *pcolor* MATLAB® command. The *multiple_contourplot.m* file displays all time-frequency plane from all the layers using the *contour* MATLAB® command. The *multiple_meshplot.m* file displays a 3-dimensional plot from all the layers using the *mesh* MATLAB® command. The *multiple_surfplot.m* file displays a 3-dimensional plot from all the layers using the *surf* MATLAB® command.

This chapter presented the QMFB theory, discussing the mathematical background for signal detection and the waveform decomposition using Wavelets, and finished with a description of the MATLAB® codes used to implement the QMFB tree. The next chapter has the most relevant work of this thesis. There, a set of different and well known LPI radar signals are processed with the QMFB tree and the output are displayed and analyzed as a feature extraction procedure. Chapter III finishes with an analysis and comparison of the different signals processing results, where the performances of the QMFB three are specified for every class of signal depending on the results obtained in the corresponding signal processing.

III. OVERVIEW OF LPI EMITTER WAVEFORMS AND PROCESSING BY THE QMFB TREE

This chapter discusses the results of processing the LPI signals from the LPIG code as described in [17]. Chapter three finishes with an analysis and comparison of the different signals processing results, where the performances of the QMFB tree are specified for every class of signal depending on the results obtained in the corresponding signal processing.

In this chapter the signals are briefly described. Next, the parameters from each signal are extracted from the QMFB results. To standardize the manner in which the signals are named, Table 1 is used. This table applies abbreviated names to the signal with numbers to identify the frequencies involved and some of the parameters such as the number of phases and code periods per phase. Table 1 shows an example of the LPIG naming convention for the collection of signals. The example shows a BPSK signal with a 2 kHz as carrier frequency, 4 kHz as sampling frequency, 13 Bits Barker Code, and 1 cycle per Barker Bit. The list of signals generated by LPIG and processed by the QMFB tree appears in Appendix B.

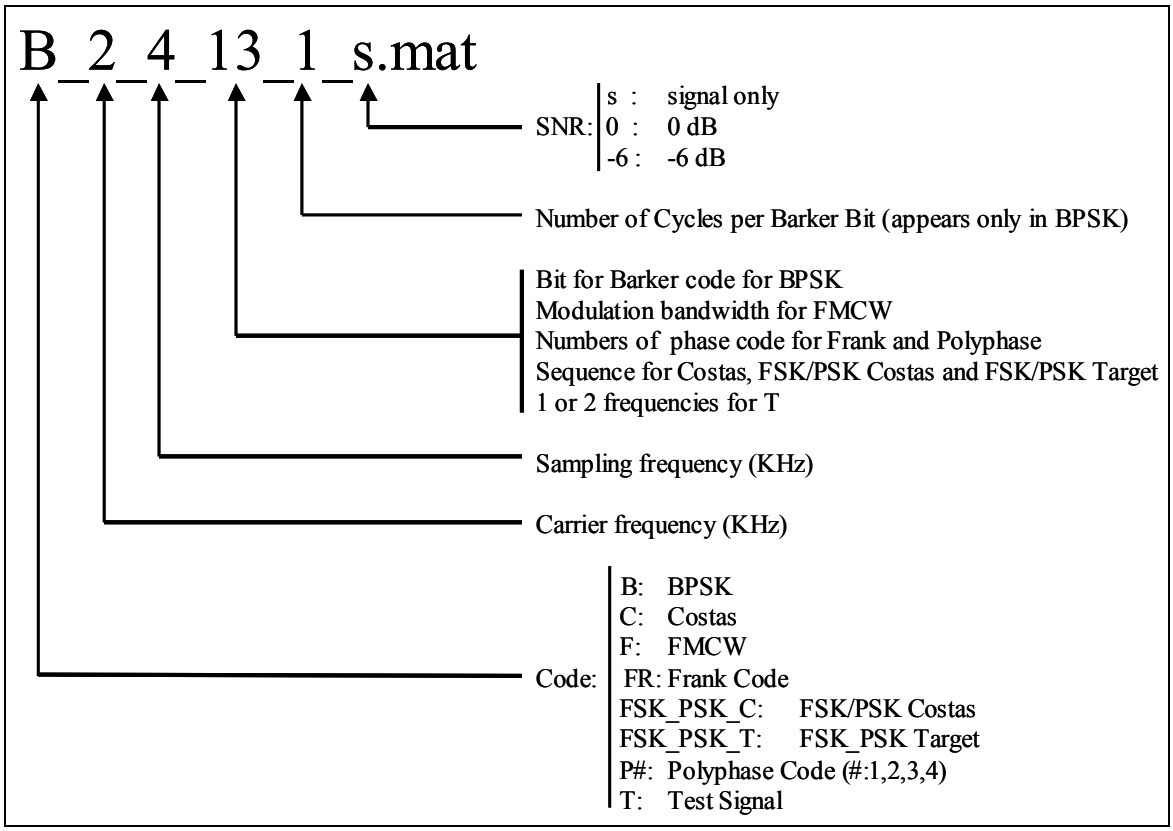

Table 1 Standard named signals from LPIG.

A. TONE TEST

To probe the type of results generated by the QMFB tree, a tone test was applied using a single frequency input. The output matrices from the various layers were examined to verify that the code *qmfb.m* is correct, and, second, to provide the frequency response information from the modified sinc filter.

1. T_1_7_1_s

The first of these tests signals uses 1 kHz tone sampled at 7 kHz with amplitude of unity and zero phases (T_1_7_1_s). The results are presented in Figure 16 showing five of the seven layers obtained by the *colorplot.m* file. Since the first and the last layers in the processing always are going to be a row data (vector, not matrix) they can not be displayed in a time-frequency diagram.

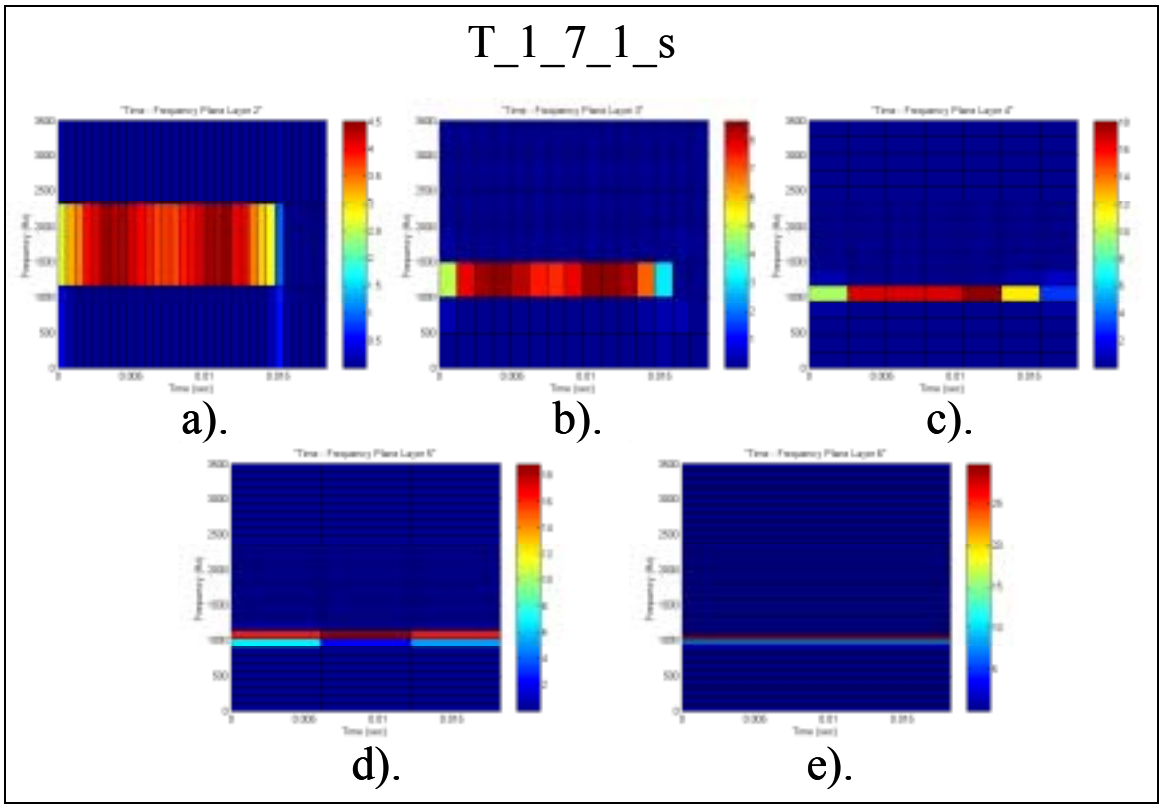

Figure 16 Time-frequency plots for T_1_7_1_s, from: a) layer 2, b) layer 3, c) layer 4, d) layer 5, e) layer 6.

In this figure, it is possible to see how each layer results in a matrix of energy values for tiles that are twice as long (in time) and half as tall (in frequency) as the tile in the previous layer. By properly comparing these matrices, it is possible to find concentrations of energy and to estimate their position and sizes with high resolution in both time and frequency. Figure 17 shows the output matrix from layer 2. The total number of layer in this example is 7 (*lay*=7) This results in very narrow tiles in time, but very wide tiles in frequency and indicates that the resolution of layer 2 is better in time than in frequency. The tiles have a resolution in frequency determined by

$$\begin{aligned}\Delta f &= \frac{f_s}{2(\text{number of columns of the output matrix at this layer}) - 1} \\ &= \frac{f_s}{2(2^{\text{number of this layer}} - 1)} = \frac{f_s}{2(\text{number of tiles in frequency})} \\ &= \frac{7000}{2(3)} = 1166.67 \text{ Hz}\end{aligned} \tag{3.1}$$

and since the sampling period of the signal T is defined as $1/f_s$, the resolution in time is determined by

$$\begin{aligned}\Delta t &= \frac{(\text{number of samples of input signal padded with zeros})(T)}{(\text{number of rows of the output matrix at this layer})\text{-}1} \\ &= \frac{(\text{number of samples of input signal padded with zeros})(1/f_s)}{(\text{number of tiles in time})} \\ &= \frac{(\text{number of samples of input signal padded with zeros})(1/f_s)}{(2^{(\text{total number of layers - number of actual layer})}-1)} \\ &= \frac{(128)(1/7000)}{31} = 0.00059 = 590\ \mu\text{s}.\end{aligned} \tag{3.2}$$

Thus, from Figure 17, the signal is contained in the frequency band 1166.67 Hz to 2333.34 Hz, and from 0 ms to 14.75 ms.

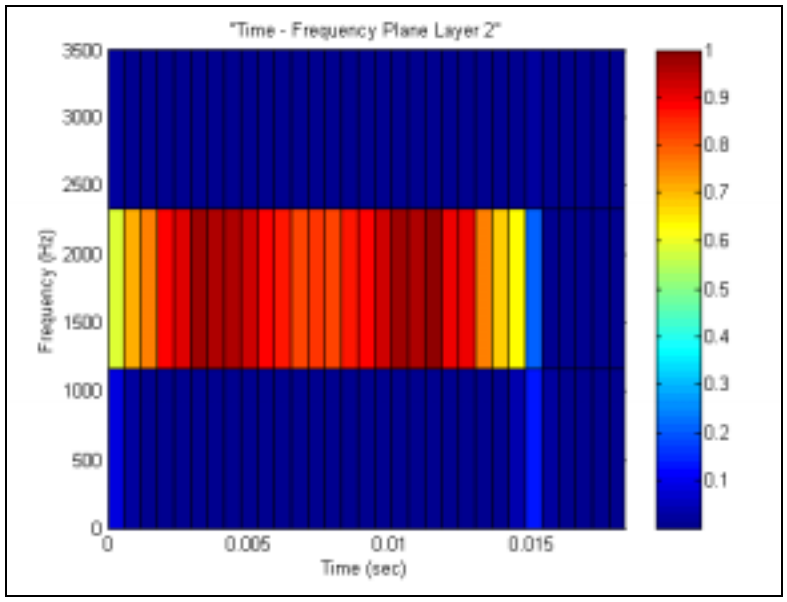

Figure 17 Time-frequency plot from layer 2 for T_1_7_1_s.

Figure 18 shows the output matrix from layer 4. The tiles have a resolution in frequency of 233.333 Hz and a resolution in time of 2.612 ms. This figure shows the signal contained in the frequency band 9333.33 Hz to 11666.7 Hz, and from 0 to 15.672 ms.

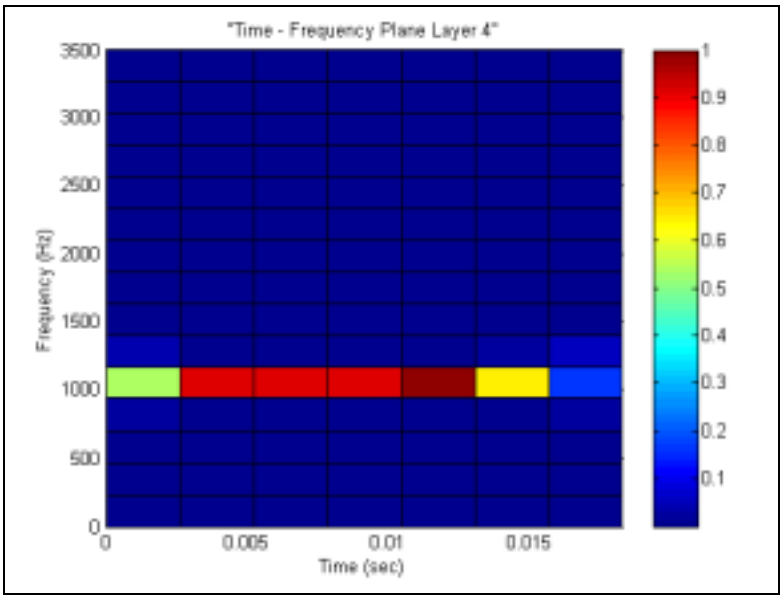

Figure 18 Time-frequency plot from layer 4 for T_1_7_1_s.

Figure 19 shows the output matrix from layer 6 with tiles being very defined in frequency. The tiles have a resolution in frequency of 55.5556 Hz and a resolution in time of 18.286 ms. Therefore, this figure shows the signal contained in the frequency band 944.445 Hz to 1055.56 Hz and from 0 to 18.286 ms. This effectively indicates that at a higher layer, the frequency resolution is higher. Thus, it could be seen that the central frequency corresponds to 1000 Hz approximately. It is important to note that the time information is not missing. The only difference is that if some more detailed specific time information is needed, it could be obtained from previous layers other than layer 6.

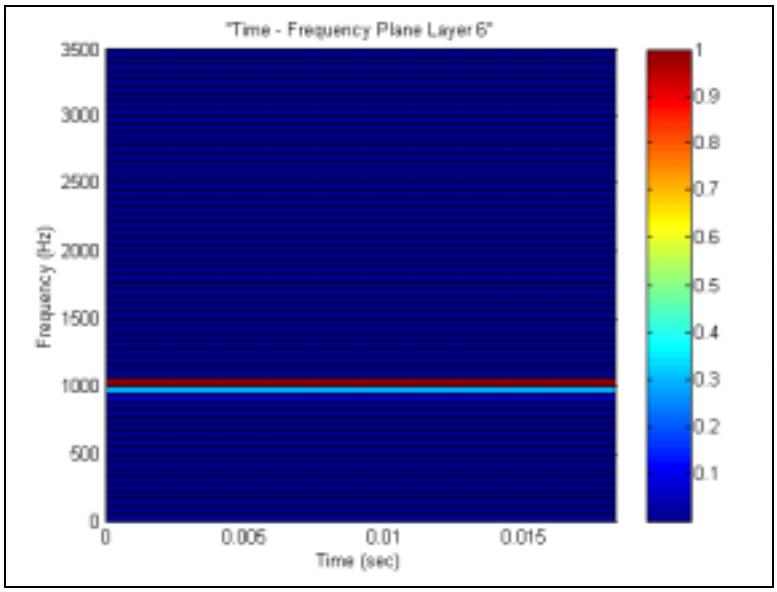

Figure 19 Time-frequency plot from layer 6 for T_1_7_1_s.

It is also instructive to look at the frequency through time. Using the *surfplot.m* file in Appendix A, the frequency-energy plots of the tone test are shown in Figure 20 to Figure 22. Figure 20 and Figure 21 show a surf plot (view from the frequency axis) from layer 2 and 4, respectively. In Figure 20, the frequency resolution is 1166.67 Hz, but clearly the energy concentrates close to 1000 Hz. In Figure 21, the frequency resolution is 233.333 Hz, been higher than the previous case.

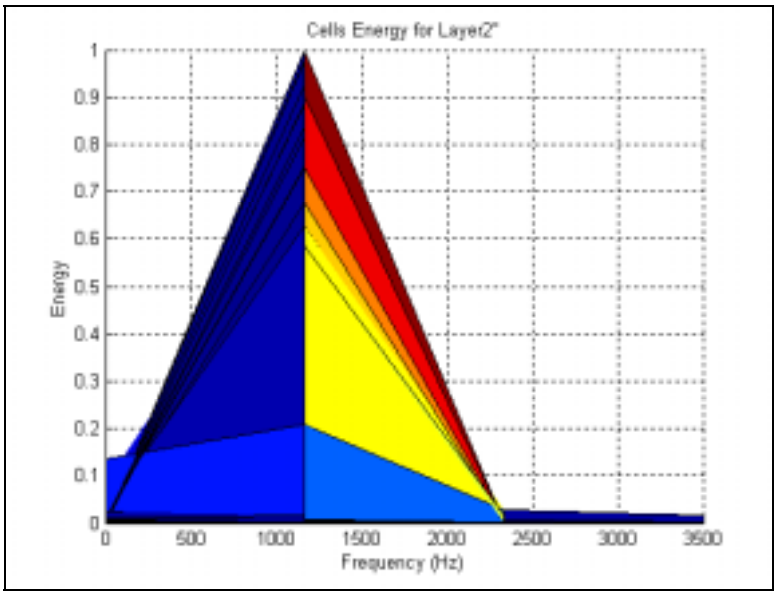

Figure 20 Frequency-energy plot from layer 2 for T_1_7_1_s.

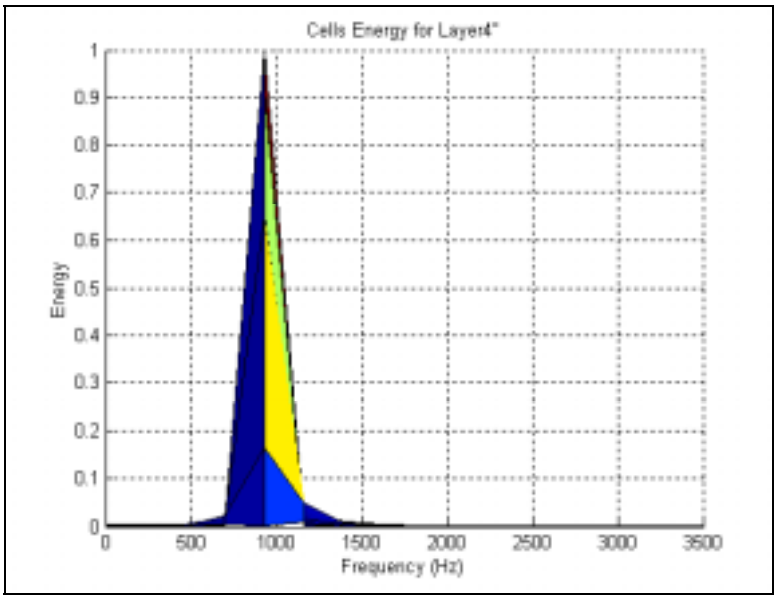

Figure 21 Frequency-energy plot from layer 4 for T_1_7_1_s.

Figure 22 shows a surf plot (view from the frequency axis) from layer 6. The frequency resolution is 55.5556 Hz, been higher than the previous cases in layers 2 and 4, which corroborates that the central frequency is 1000 Hz for the test signal. The last three figures reveal that, the higher the layer, the higher the resolution in frequency. Layer 6 gives the best resolution and is shown as a 3-dimensional surf plot in Figure 23 with a frequency resolution of 55.5556 Hz.

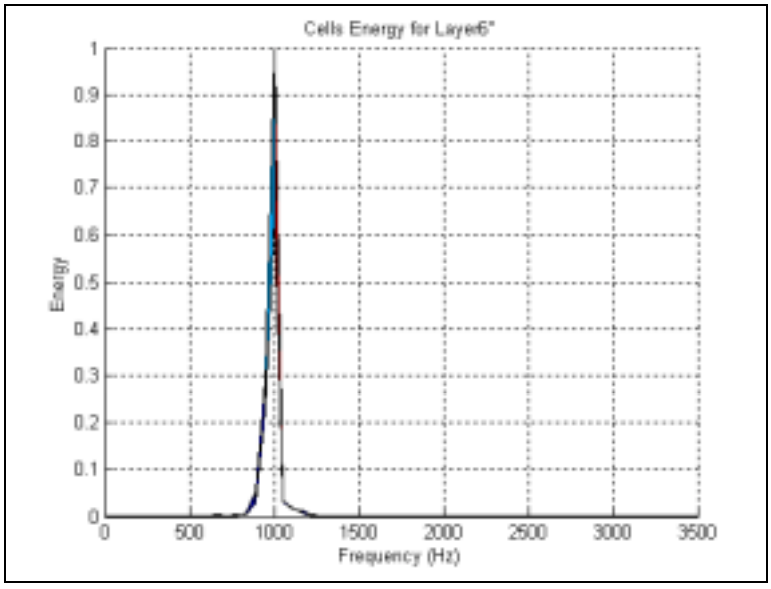

Figure 22 Frequency-energy plot from layer 4 for T_1_7_1_s.

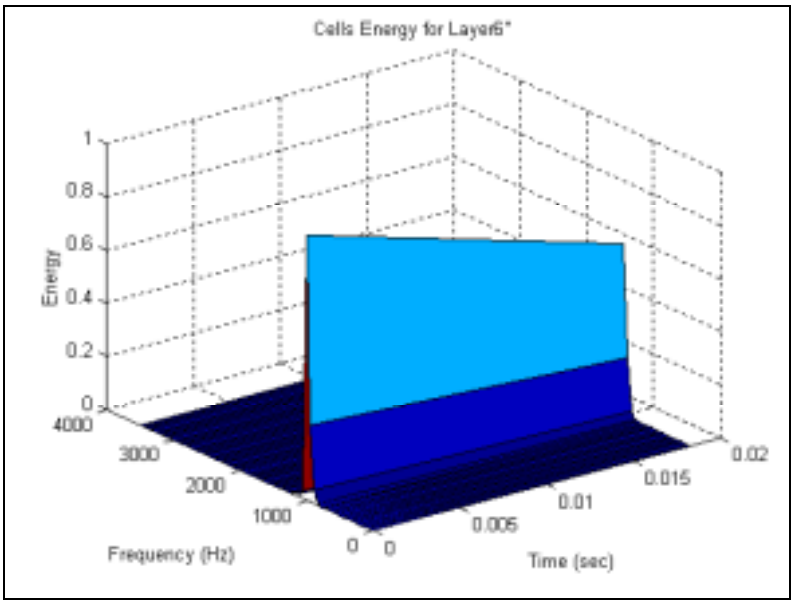

Figure 23 Surf plot from layer 6 for T_1_7_1_s.

Table 2 presents a summary of the signal processing, showing the relevant parameters extracted from the QMFB layers 2, 4 and 6.

T_1_7_1_s	Frequency Resolution	Time Resolution	Comment
Layer 2	1166.67 Hz	0.000590 ms	
Layer 4	233.333 Hz	2.612 ms	
Layer 6	55.5556	18.286 ms	
Carrier Frequency	1 kHz	-	
Sampling Frequency	7 kHz	-	
SNR	Only Signal	-	

Table 2 Signal processing summary for T_1_7_1_s.

2. T_12_7_2_s

A second signal test is applied to the QMFB tree. The signal was generated with two tones, 1 kHz and 2 kHz, sampled at 7 kHz with amplitude of unity and zero phase (T_12_7_2_s) are used for this second test. The amount of layers obtained when processing the signal with the QMFB, as it was seen in Chapter II Section C.1, depends on the length of the signal to process, and corresponds to the power of 2 that determine this length after padding with zeros if it is necessary. Since the first and the last layers in the processing always are going to be a row data (vector, not matrix) they can not be displayed in a time-frequency diagram. Thus, the rest of the layers obtained are presented in

Figure 24. In this figure, it is possible to see how each layer outputs a matrix of energy values for tiles with an increment in frequency resolution as the layers increase. By properly comparing these matrices, it is possible to find concentrations of energy, and estimate their position and sizes with high resolution in both time and frequency.

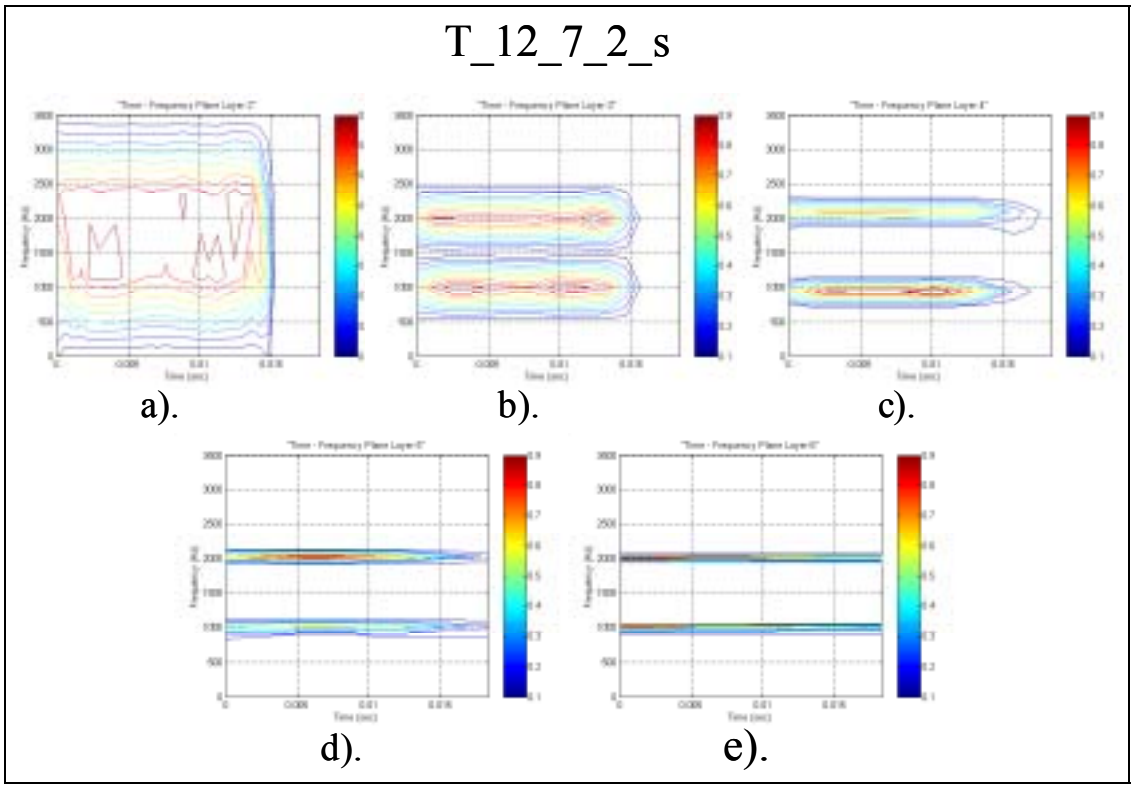

Figure 24 Time-frequency plots for T_12_7_2_s, from: a) layer 2, b) layer 3, c) layer 4, d) layer 5, e) layer 6.

Figure 25 shows the output matrix from layer 2. The resolution in time is 0.59 ms being better than the resolution in frequency that is 1166.67 Hz. In fact, the signal covers a wide band in frequency.

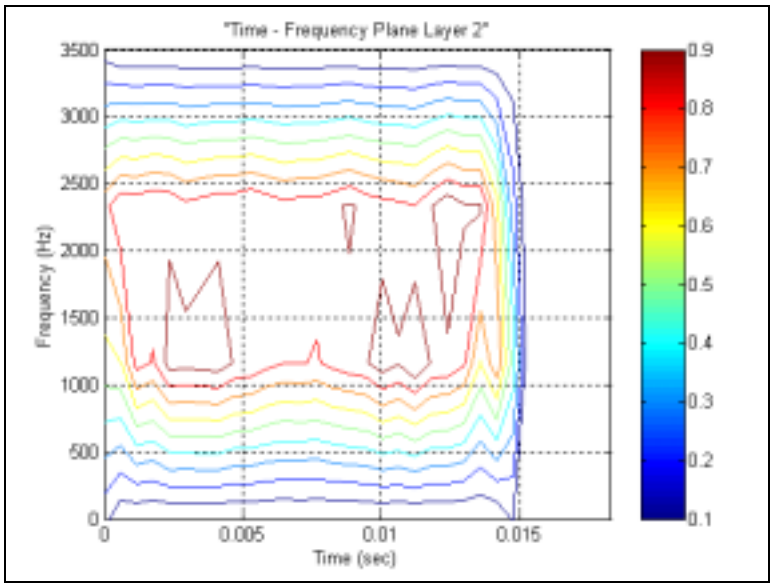

Figure 25 Contour plot from layer 2 for T_12_7_2_s.

Figure 26 shows the output matrix from layer 4. The resolution in frequency is 233.333 Hz being higher than in layer 2. The plot demonstrates that the carrier frequency is close to the test signal, which are 1000 Hz and 2000 Hz, but still with some bandwidth, making it not as precise as desired. However, Figure 27 shows the output matrix from layer 6 with frequency resolution of 55.5556 Hz and there the central frequencies corresponds to 1000 Hz and 2000 Hz.

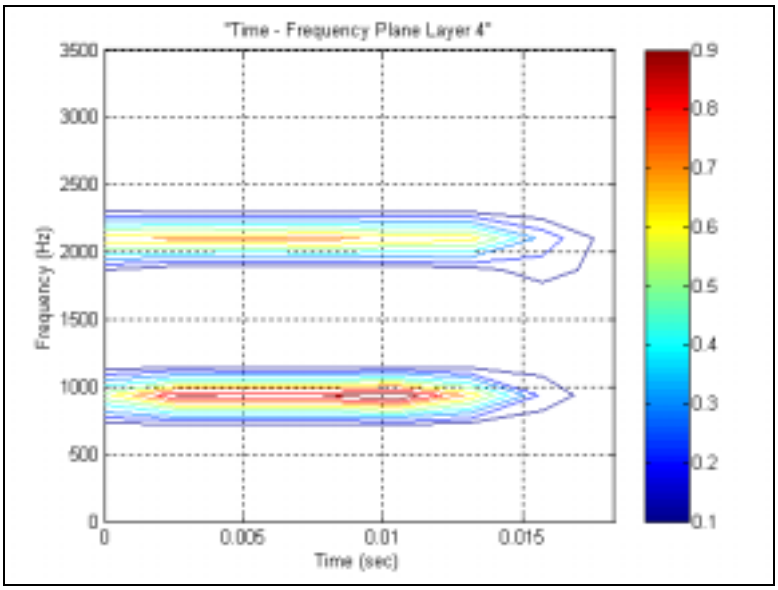

Figure 26 Contour plot from layer 4 for T_12_7_2_s.

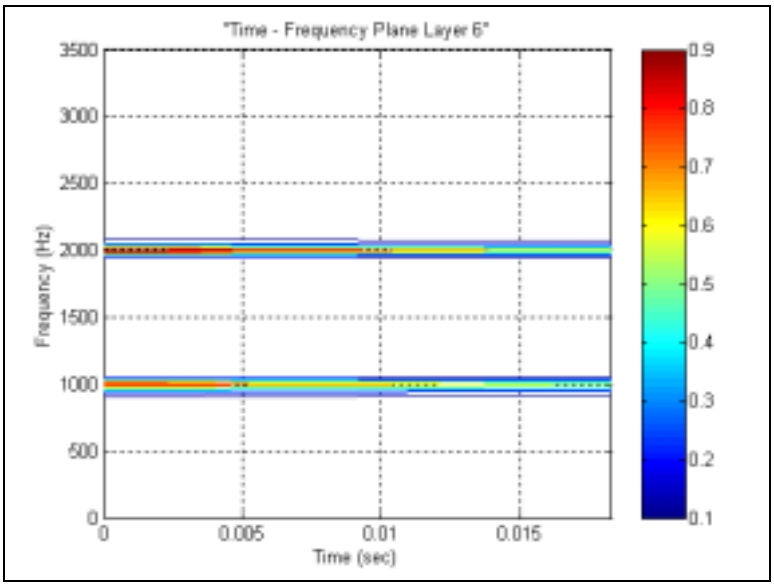

Figure 27 Contour plot from layer 6 for T_12_7_2_s.

As with the previous test signal, the frequency through time is considered. Using the *surfplot.m* file, a frequency-energy plot of the tone test signal is shown in Figure 28 to Figure 30. Figure 28 and Figure 29 show a surf plot (view from the frequency axis) from layer 2 and 4, respectively. In Figure 28, the frequency resolution is 1166.67 Hz (not high enough), but clearly the energy concentrates between 1000 Hz and 2000 Hz. In Figure 29, the frequency resolution is 233.333 Hz (higher than the previous case), approximating the both frequencies at 1000 Hz and 2000 Hz.

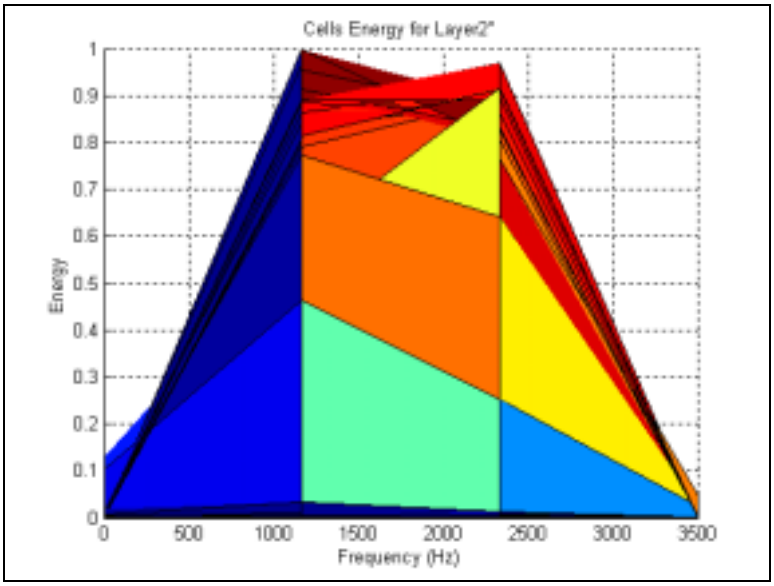

Figure 28 Frequency-energy plot from layer 2 for T_12_7_2_s.

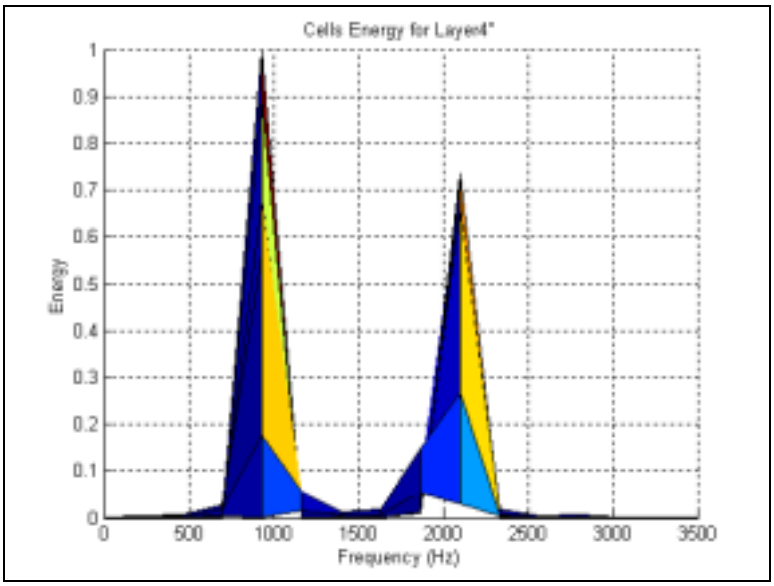

Figure 29 Frequency-energy plot from layer 4 for T_12_7_2_s.

Figure 30 shows a surf plot (view from the frequency axis) from layer 6. The frequency resolution is 55.5556 Hz and verifies that the central frequencies are 1000 Hz and 2000 Hz for the test signal.

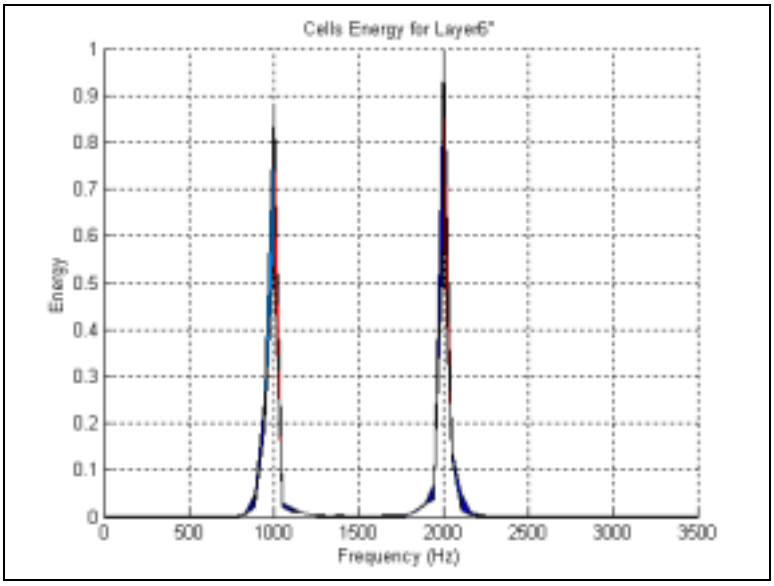

Figure 30 Frequency-energy plot from layer 6 for T_12_7_2_s.

The last three figures demonstrate that the higher the layer, the higher the resolution in frequency. Thus, the best resolution is that presented by layer 6. Figure 31 shows a 3-dimensional surf plot from layer 6 with a frequency resolution of 55.5556 Hz.

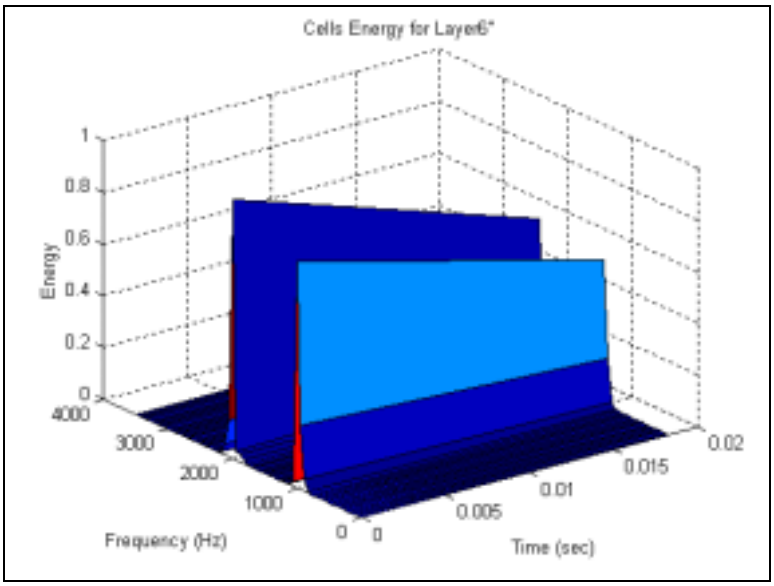

Figure 31 Surf plot from layer 6 for T_12_7_2_s.

Table 3 presents a summary of the signal processing, showing the relevant parameters extracted from the QMFB layers 2, 4 and 6.

T_1_7_1_s	Frequency Resolution	Time Resolution	Comment
Layer 2	1166.67 Hz	0.000590 ms	
Layer 4	233.333 Hz	2.612 ms	
Layer 6	55.5556	18.286 ms	
Carrier Frequency	1 and 2 kHz	-	
Sampling Frequency	7 kHz	-	
SNR	Only Signal	-	

Table 3 Signal processing summary for T_12_7_2_s.

Up to this point, two different methods were used to show the output matrix. They are the color and contour plots. From both it is possible to conduct the feature extraction, but the selected plot to be shown in the following signal processing will depend on which will display a better resolution from any layer.

B. BINARY PHASE SHIFT KEYING (BPSK)

1. Brief Description

Phase shift keying (PSK) is a form of angle-modulated, constant-amplitude digital modulation. PSK is similar to conventional phase modulation except that with PSK the input signal is a binary digital signal and only a limited number of output phases are possible. BPSK is not a technique employed in radar modulation, but is a well-known LPI signal used in communication, thus constitutes itself as an excellent test signal in evaluating the performance of the QMFB processing.

With binary phase shift keying (BPSK), two output phases are possible for a single carrier frequency (binary, meaning 2). One output phase represents logic 1 and the other a logic 0. As the input digital signal changes state, the phase of the output carrier shifts between two angles that are 180° out of phase. BPSK is a form of suppressed-carrier, square-wave modulation of a continuous wave (CW) signal [14].

In a BPSK transmitter, the modulator receives two input signals, the carrier and the binary data. If +1V is assigned to a logic 1 and –1V is assigned to a logic 0, the input carrier $(\sin \omega_c t)$ is multiplied by either a + or –1. Consequently, the output signal is either $+1 \sin \omega_c t$ or $-1 \sin \omega_c t$. The first represents a signal that is in phase with the reference oscillator while the latter is a signal 180° out of phase with the reference oscillator. Each time the input logic condition changes, the output phase changes. Mathematically, the output of a BPSK modulator is proportional to

$$BPSK\ output = [\sin(2\pi f_a t)][\sin(2\pi f_c t)] \tag{3.3}$$

where f_a = maximum fundamental frequency of binary input (Hertz) and f_c = reference carrier frequency (Hertz). Solving for the trig identity for the product of two sine functions

$$\text{BPSK output} = \frac{1}{2}\cos[2\pi(f_c - f_a)t] - \frac{1}{2}\cos[2\pi(f_c + f_a)t]. \tag{3.4}$$

Figure 32 below shows a basic block diagram of the transmitter design. The signal $x(t)$ is a sinusoidal CW. After sampling at the Nyquist rate, the modulate signal is created by adding a n-bit Barker code. This code has been widely used because of its low-side lobes at zero Doppler. Once the signal has been modulated, white Gaussian noise is added at the desired level.

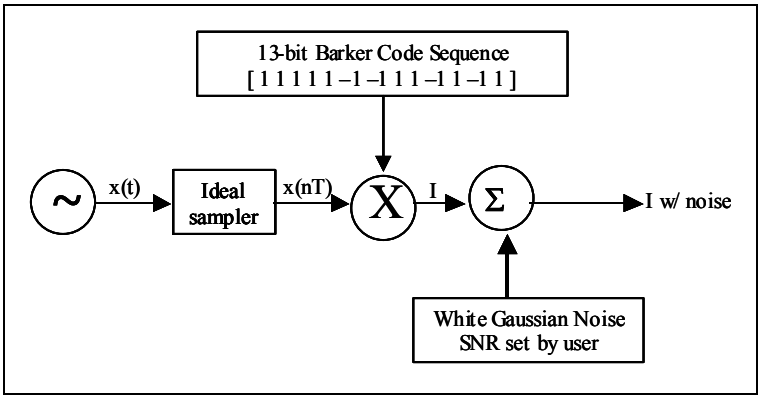

Figure 32 BPSK transmitter block diagram.

The modulating signal waveform is represented by the dashed lines in the upper graph in Figure 33. The number of periods per Barker Bit is set to one, signifying that one full period of the sampled signal fits within one bit of the 13-bit Barker code. The first five bits of the Barker code are +1 and the next 2 bits are –1, so five full periods under the first +1 portion of the modulating waveform, 2 full periods under the –1 portion of the modulating waveform, and so forth are seen. In this figure, only the first 12 bits of the 13-bit Barker code are represented. Table 4 shows Barker codes for 7, 11, and 13 bits.

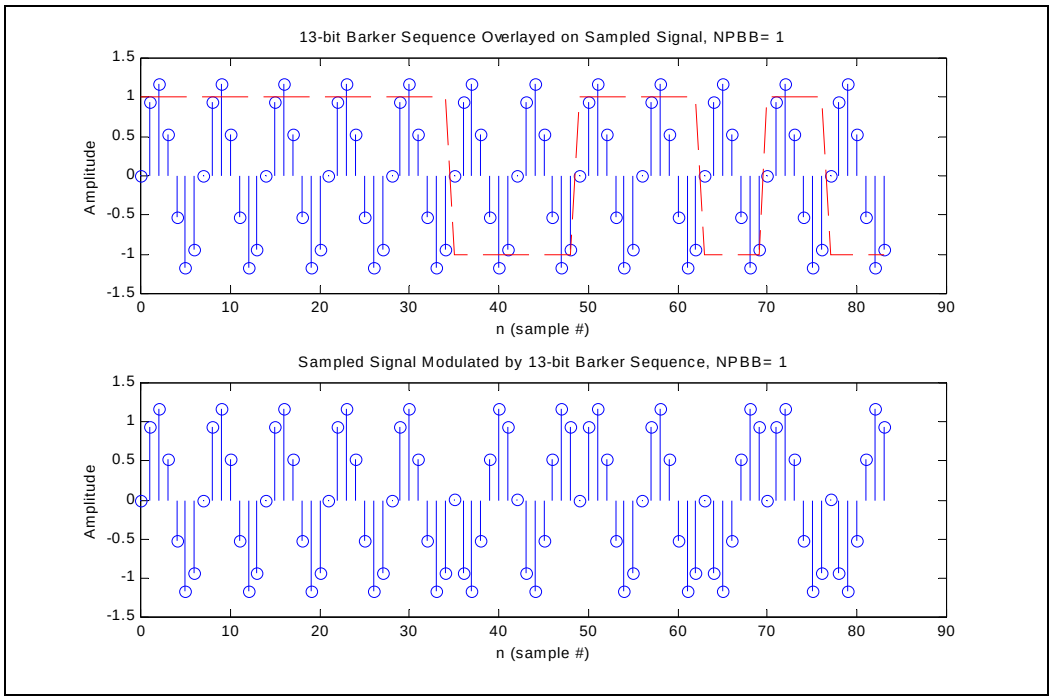

Figure 33 Sampled signal (upper plot in blue color), modulating signal (upper plot, in red color) and modulated signal (lower plot, in blue color).

Number of bits	Barker Code
7	+ + + – – + –
11	+ + + – – – + – – + –
13	+ + + + + – – + + – + – +

Table 4 Barker Code for 7, 11 and 13 Bits.

2. Processing BPSK Signals with QMFB Tree

The BPSK signals to be worked with are given in Table 5. All the signals were generated with a carrier frequency (f_C) of 1000 Hz and sampling frequency (f_S) of 7000 Hz. The number of bits per Barker code, the number of cycles per bit (*cpb*), and the SNR are variable parameters.

	BPSK	Number of bits per Barker code (Bits)	Number of cycles per bit (cpb)	SNR
1	B_1_7_7_1_s	7	1	Signal Only
2	B_1_7_7_1_0	7	1	0 dB
3	B_1_7_7_1_–6	7	1	–6 dB
4	B_1_7_11_1_s	11	1	Signal Only
5	B_1_7_11_1_0	11	1	0 dB
6	B_1_7_11_1_–6	11	1	–6 dB
7	B_1_7_7_5_s	7	5	Signal Only
8	B_1_7_7_5_0	7	5	0 dB
9	B_1_7_7_5_–6	7	5	–6 dB
10	B_1_7_11_5_s	11	5	Signal Only
11	B_1_7_11_5_0	11	5	0 dB
12	B_1_7_11_5_–6	11	5	–6 dB

Table 5 BPSK signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 5), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. B_1_7_7_5_s

This BPSK signal was generated with the parameters described in Table 6.

The code period of the BPSK signal is

$$T_C = \frac{(\text{Number of cycles per Barker bit})(\text{Number of bits per Barker code})}{\text{Carrier Frequency}}$$
$$T_C = \frac{(\text{cpb})(\text{Bits})}{f_C} = \frac{(5)(7)}{1000} = 0.035 \text{ s} = 35 \text{ ms}. \quad (3.5)$$

The bandwidth of the signal depends on the cycles per bit (or chirp), given by the LPIG code in the generation of the signal. Thus,

$$BW = \frac{f_C}{\text{cpb}} = \frac{1000 \text{ Hz}}{5 \text{ cycles per bit}} = 200 \text{ Hz} = 0.2 \text{ KHz}. \quad (3.6)$$

Now, and according to the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB tree to get the different output layers. In this manner, Figure 34 was created which shows the output matrix from

layer 6 in a contourplot with a frequency resolution of 55.55 Hz and a time resolution of 9.677 ms. The values of the carrier frequency, bandwidth, and code period are extracted. Figure 35 shows the output matrix from layer 2, there it is noted the phase shifts in the signal, the Barker code sequence, and corroborate the code period.

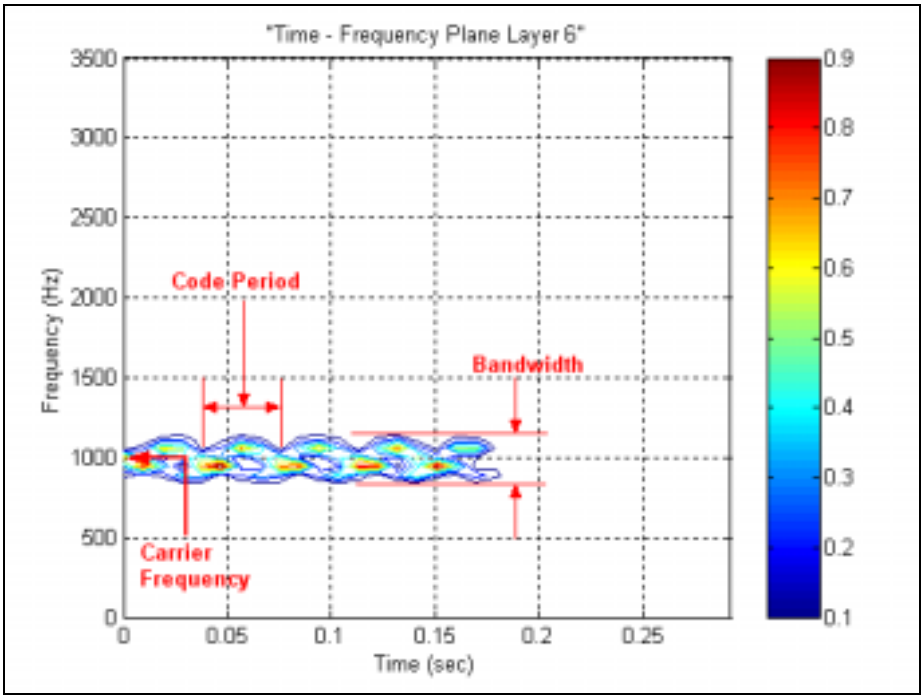

Figure 34 Output matrix from layer 6 of B_1_7_7_5_s (contourplot).

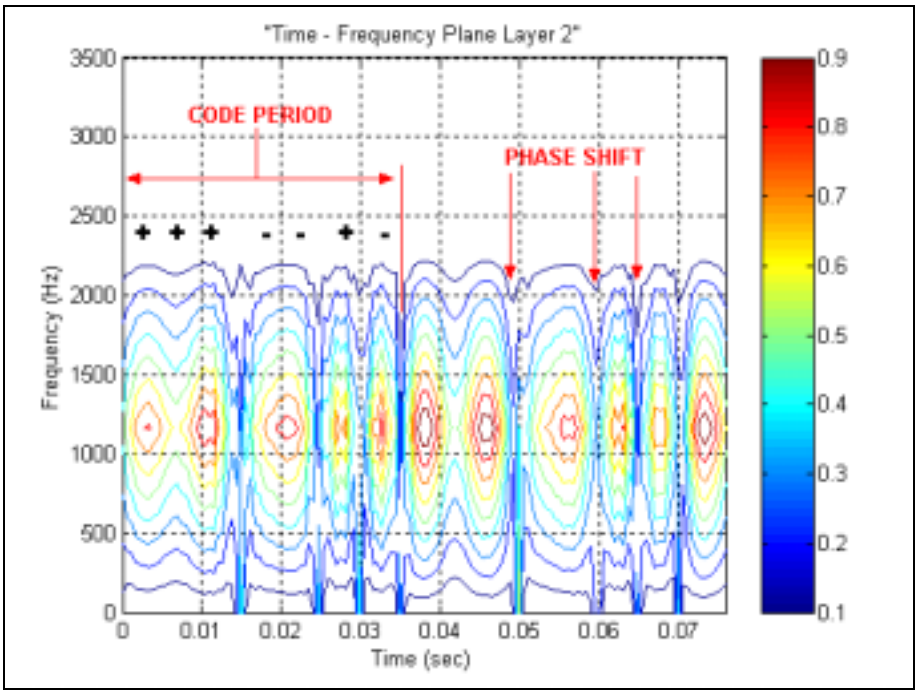

Figure 35 Output matrix from layer 2 of B_1_7_7_5_s (contourplot).

Table 6 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

B_1_7_7_5_s Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	given
Bits	7 bits	7	
cpb	5	-	
Bandwidth	0.2 kHz	0.222 kHz	0.022 kHz error
Code Period	35 ms	38.708 ms	3.708 ms error
SNR	Only Signal	-	

Table 6 Signal processing summary for B_1_7_7_5_s.

b. B_1_7_7_5_0

This BPSK signal was generated with the parameters described in Table 7. The different output layers were obtained conducting the signal processing with the QMFB tree. Figure 36 presents the output matrix from layer 6 with a frequency resolution of 55.55 Hz and a time resolution of 9.677 ms. There the values of the carrier frequency and the bandwidth are extracted matching the signal values. Since it still is possible to find a pattern or a frequency in the repetition of tiles and the resolution in time

is acceptable, it was not necessary to examine in previous layers to find the code period. Consequently, this parameter was extracted from the same figure, matching the value in (3.5).

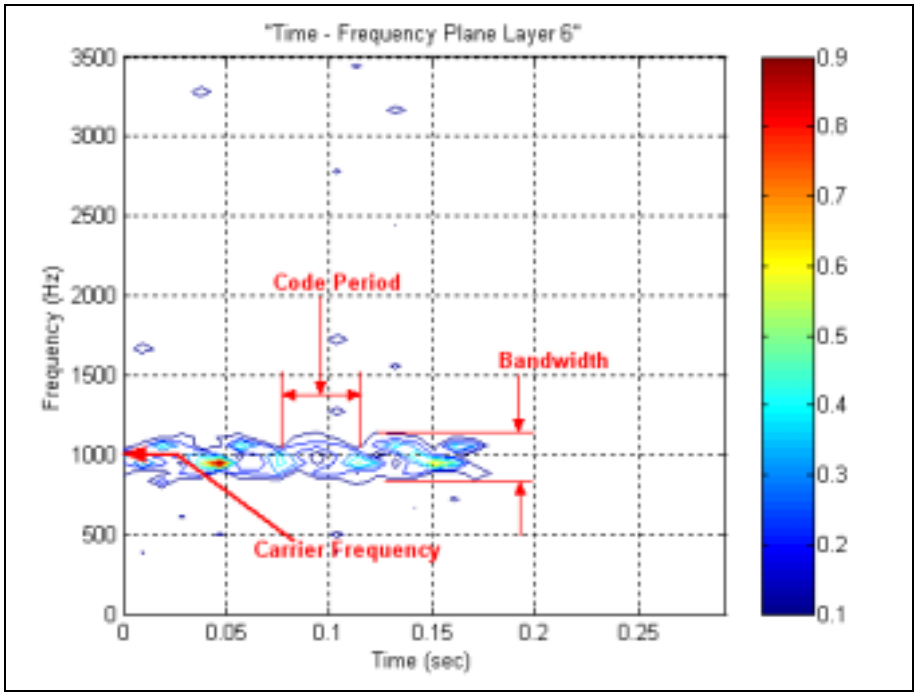

Figure 36 Output matrix from layer 6 of B_1_7_7_5_0 (contourplot).

Table 7 presents a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

B_1_7_7_5_0 Parameters	**Generation**	**Detection**	**Comment**
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	given
Bits	7 bits	-	
cpb	5	-	
Bandwidth	0.2 kHz	0.333 kHz	0.133 kHz error
Code Period	35 ms	38.708 ms	3.708 ms error
SNR	0 dB	-	

Table 7 Signal processing summary for B_1_7_7_5_0.

c. B_1_7_7_5_–6

This BPSK signal was generated with the parameters described in Table 8. Processing the signal with the QMFB Tree to find the different output layers, Figure 37 was obtained, which show the output matrix from layer 6 with a frequency resolution of 55.55 Hz and a time resolution of 9.677 ms. There the values of the carrier frequency and the bandwidth are extracted. The frequency matches the signal value, but the bandwidth gives a higher value. It was not possible to extract the code period from the plot due to the noise in the signal.

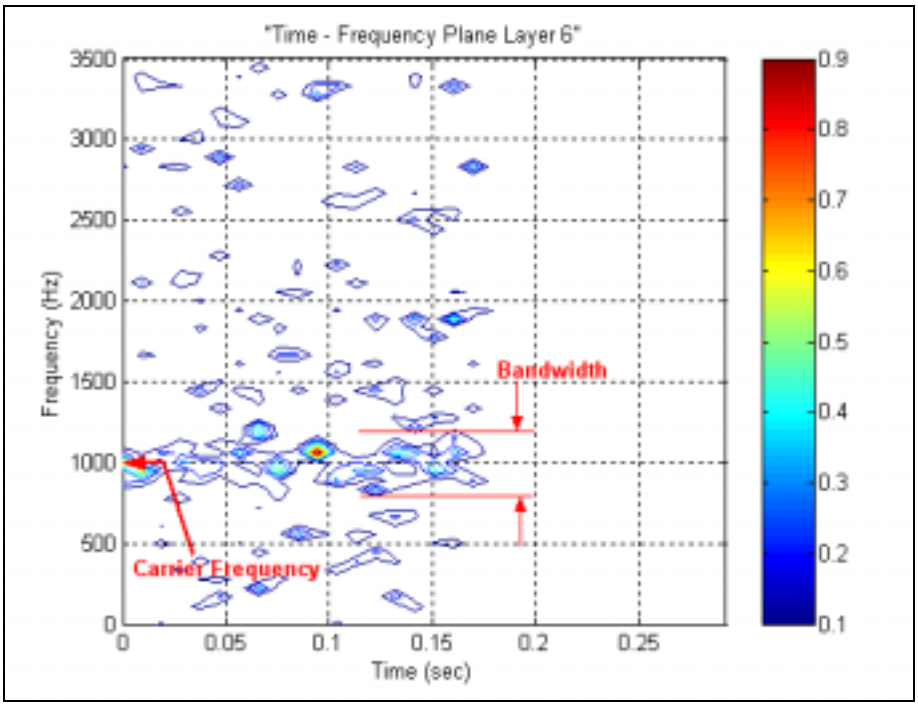

Figure 37 Output matrix from layer 6 of B_1_7_7_5_–6 (contourplot).

Table 8 presents a summary of the signal processing indicating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

B_1_7_7_5_–6 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	given
Bits	7 bits	-	
cpb	5	-	
Bandwidth	0.2 kHz	0.444 kHz	0.244 kHz error
Code Period	35 ms	-	
SNR	–6 dB	-	

Table 8 Signal processing summary for B_1_7_7_5_–6.

C. FREQUENCY MODULATED CONTINUOUS WAVE (FMCW)

1. Brief Description

Linear frequency modulation in addition to being an LPI radar technique has the added advantage that the modulation is readily compatible with solid-state transmitters. The most popular modulation is the triangular modulation of a Frequency Modulated Continuous Wave (FMCW). The linear FMCW emitter uses a continuous 100% duty-cycle waveform so that both the target range and the Doppler information can be measured unambiguously while maintaining a low probability of intercept. The FMCW waveform represents the best use of the output power available from solid-state devices. Linear FMCW is easier to implement than phase code modulation as long as there is not strict demand on linearity over the modulation bandwidth [1].

The linear frequency modulated triangular waveform and the Doppler shifted signal are shown in Figure 38. The triangular modulation consists of two linear frequency modulation sections with positive and negative slopes. With this configuration, the range and Doppler frequency of the detected target can be extracted unambiguously by taking the sum and the difference of the two beat frequencies [1].

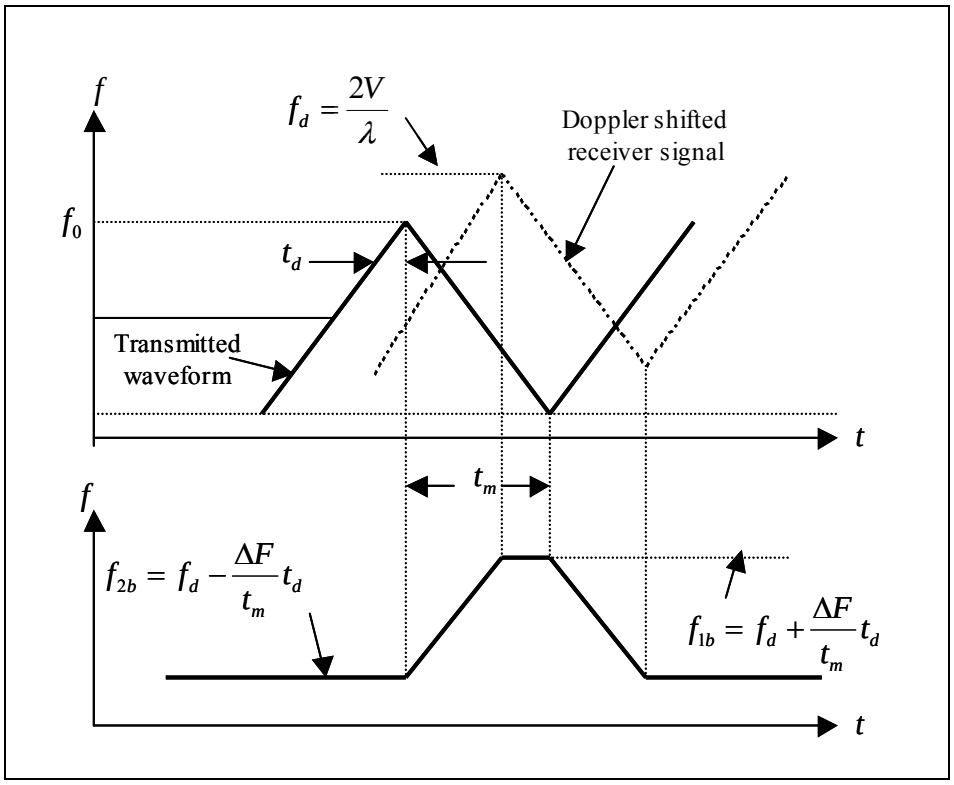

Figure 38 Linear frequency modulated triangular waveform and the Doppler shifted signal.

The frequency of the transmitted signal for the first section is

$$f_1(t) = f_0 - \frac{\Delta F}{2} + \frac{\Delta F}{t_m} t \tag{3.7}$$

for $0 < t < t_m$ and zero elsewhere. Here f_0 is the RF carrier, ΔF is the transmitted modulation bandwidth, and t_m is the modulation period. The rate of the frequency change or chirp rate $\dot{F}$ is

$$\dot{F} = \frac{\Delta F}{t_m}. \tag{3.8}$$

The phase of the transmitted RF signal is

$$\phi_1(t) = 2\pi \int_0^t f_1(x)\, dx. \tag{3.9}$$

Assuming that $\phi_o = 0$ at $t = 0$,

$$\phi_1(t) = 2\pi\left[\left(f_o - \frac{\Delta F}{2}\right)t + \frac{\Delta F}{2t_m}t^2\right] \tag{3.10}$$

for $0 < t < t_m$. The transmit signal is given by

$$s_1(t) = a_o \sin 2\pi\left[\left(f_o - \frac{\Delta F}{2}\right)t + \frac{\Delta F}{2t_m}t^2\right]. \tag{3.11}$$

The frequency of the transmitted waveform of the second section is

$$f_2(t) = f_0 + \frac{\Delta F}{2} - \frac{\Delta F}{t_m}t \tag{3.12}$$

for $0 < t < t_m$. The transmitted baseband signal is given by

$$s_2(t) = a_o \sin 2\pi\left[\left(f_o + \frac{\Delta F}{2}\right)t - \frac{\Delta F}{2t_m}t^2\right]. \tag{3.13}$$

2. Processing FMCW Signals with QMFB Tree

The FMCW signals to be worked with are given in Table 9. All the signals were generated with a carrier frequency (f_C) of 1000 Hz and sampling frequency (f_S) of 7000 Hz. The modulation bandwidth (Hz), the modulation period (ms), and the SNR are variable parameters.

	FMCW	Modulation Bandwidth	Modulation Period	SNR
1	F_1_7_250_20_s	250 Hz	20 ms	Signal Only
2	F_1_7_250_20_0	250 Hz	20 ms	0 dB
3	F_1_7_250_20_–6	250 Hz	20 ms	–6 dB
4	F_1_7_250_30_s	500 Hz	30 ms	Signal Only
5	F_1_7_250_30_0	500 Hz	30 ms	0 dB
6	F_1_7_250_30_–6	500 Hz	30 ms	–6 dB
7	F_1_7_500_20_s	250 Hz	20 ms	Signal Only
8	F_1_7_500_20_0	250 Hz	20 ms	0 dB
9	F_1_7_500_20_–6	250 Hz	20 ms	–6 dB
10	F_1_7_500_30_s	500 Hz	30 ms	Signal Only
11	F_1_7_500_30_0	500 Hz	30 ms	0 dB
12	F_1_7_500_30_–6	500 Hz	30 ms	–6 dB

Table 9 FMCW signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 9), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. F_1_7_500_20_s

This FMCW signal was generated with the parameters described in Table 10. Therefore, and according with the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB Tree to obtain the different output layers. Thus, Figure 39 was created, which show the output matrix from layer 5 with a frequency resolution of 112.903 Hz and a time resolution of 4.762 ms. There, the values of the carrier frequency, the modulation bandwidth, and the modulation period are extracted, giving almost the same values than the original signal.

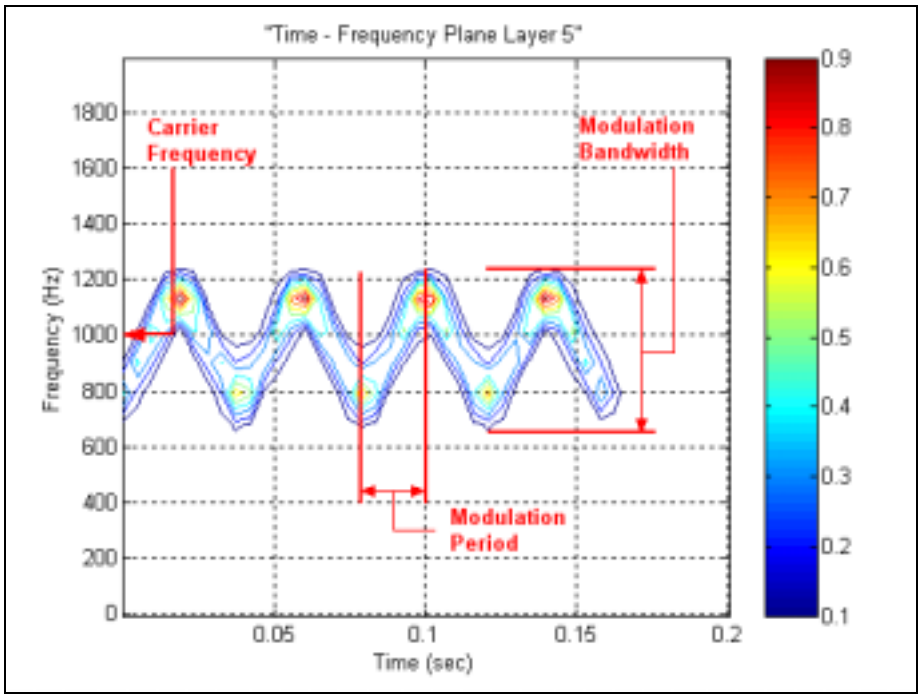

Figure 39 Output matrix from layer 5 of F_1_7_500_20_s (contourplot).

Table 10 show a summary of the signal processing showing a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

F_1_7_500_20_s Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1016.13 kHz	16.13 Hz error
Sampling Frequency	7 kHz	7 kHz	given
Modulation Band-width	500 Hz	564.515 Hz	64.515 Hz error
SNR	Only Signal	-	
Modulation Period	20 ms	23.81 ms	3.81 ms error

Table 10 Signal processing summary for F_1_7_500_20_s.

b. F_1_7_500_20_0

This FMCW signal was generated with the parameters described in Table 11. Figure 40 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 5 when conducting the processing with the QMFB tree with a frequency resolution of 112.903 Hz and a time resolution of 4.762 ms. There the values of the carrier frequency, the modulation bandwidth, and the modulation period are extracted, giving almost the same values than the original signal.

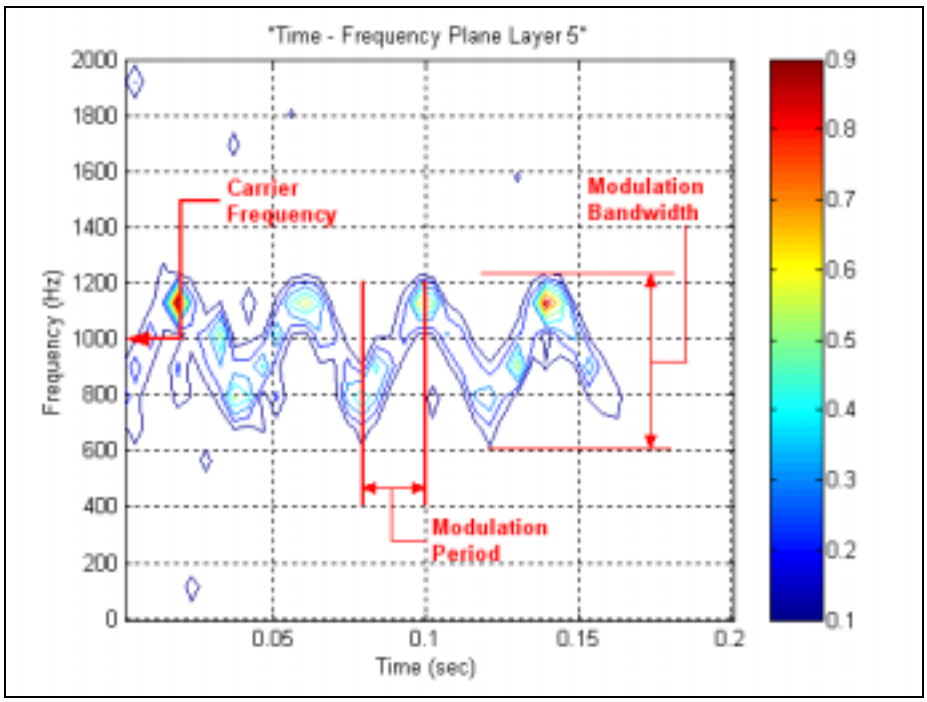

Figure 40 Output matrix from layer 5 of F_1_7_500_20_0 (contourplot).

Table 11 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

F_1_7_500_20_0 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1016.13 kHz	16.13 Hz error
Sampling Frequency	7 kHz	7 kHz	given
Modulation Band-width	500 Hz	677.418 Hz	177.418 Hz error
SNR	0 dB	-	
Modulation Period	20 ms	23.81 ms	3.81 ms error

Table 11 Signal processing summary for F_1_7_500_20_0.

c. F_1_7_500_20_–6

This FMCW signal was generated with the parameters described in Table 12. Figure 41 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 55.5556 Hz and a time resolution of 9.677 ms. There the values of the carrier frequency and the modulation bandwidth are extracted. The extracted carrier frequency matched the actual value; the extracted modulation bandwidth was in error by 55.556 Hz. The modulation period could not have been extracted due to the low SNR.

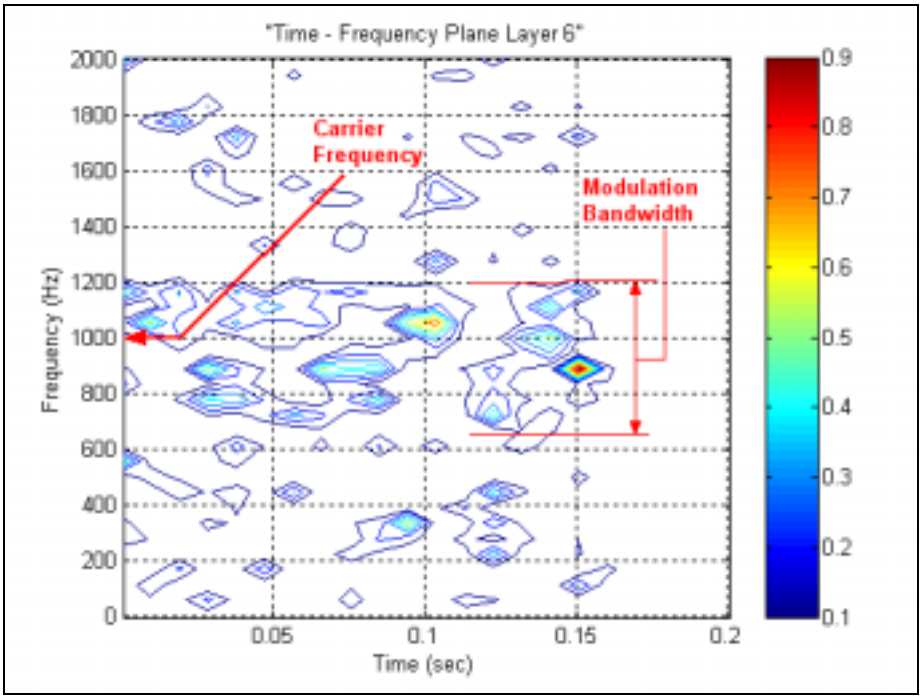

Figure 41 Output matrix from layer 6 of F_1_7_500_20_–6 (contourplot).

Table 12 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

F_1_7_500_20_–6 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	given
Modulation Bandwidth	500 Hz	555.556 Hz	55.556 Hz error
SNR	–6 dB	-	
Modulation Period	20 ms	-	

Table 12 Signal processing summary for F_1_7_500_20_–6.

D. FRANK CODES

1. Brief Description

Frank codes are a family of polyphase codes that are closely related to the Chirp and Baker codes and have been used successfully in LPI radars signals. In the case of a step approximation to linear frequency modulation with N frequency steps, N samples per frequency are obtained and *(N)(N)* samples result. That is, the Frank codes has a length N^2 which is also the corresponding pulse compression ratio. In the case of single sideband detection, the result is the Frank code [1]. For example, if the local oscillator is at the start of the sweep of the step approximation to linear frequency waveform, the first N samples of the polyphase code are 0 phase. The second N samples start with 0 phase and increase with phase increments of $(2\pi \text{ deg}/ N)$ from sample to sample, the third group of N samples starts with 0 phase and increase with $(2\pi \text{ deg}/ N)$ degrees increments from sample to sample and so on. Figure 42 shows the relationship between the index in the matrix and its phase shift for N^2=16. If i is the number of the sample in a given frequency and j is the number of the frequency, the phase of the i-th sample of the j-th frequency is

$$\phi_{i,j} = \frac{2\pi}{N}(i-1)(j-1) \tag{3.14}$$

where $i = 1, 2, ..., N$, and $j = 1, 2, ..., N$ [1]. The Frank code has the highest phase increments from sample to sample in the center of the code. The Frank polyphase code can also be described by a matrix as follows

$$\begin{bmatrix} 0 & 0 & 0 & ... & 0 \\ 0 & 1 & 2 & ... & (N-1) \\ 0 & 2 & 4 & ... & 2(N-1) \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & (N-1) & 2(N-1) & ... & (N-1)^2 \end{bmatrix} \tag{3.15}$$

where the numbers represent multiplying coefficients of the basic phase angle $2\pi / N$ [1].

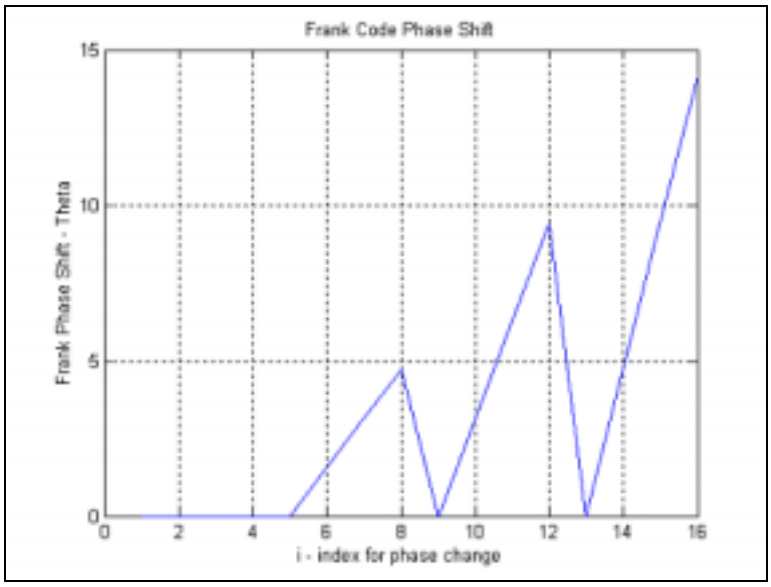

Figure 42 Phase relationship between the index in the matrix and its phase shift for N^2=16.

2. Processing Frank Code Signals with QMFB Tree.

The Frank code signals to be worked with are given in Table 13. All the signals were generated with a carrier frequency (f_C) of 1000 Hz and sampling frequency (f_S) of 7000 Hz. The number of phases (*N*), the number of cycles per phase (*cpp*) and the SNR are variable parameters.

	FRANK	Number of code phases N	Number of cycles per phase cpp	SNR
1	FR_1_7_4_1_s	4	1	Signal Only
2	FR_1_7_4_1_0	4	1	0 dB
3	FR_1_7_4_1_–6	4	1	–6 dB
4	FR_1_7_4_5_s	4	5	Signal Only
5	FR_1_7_4_5_0	4	5	0 dB
6	FR_1_7_4_5_–6	4	5	–6 dB
7	FR_1_7_8_1_s	8	1	Signal Only
8	FR_1_7_8_1_0	8	1	0 dB
9	FR_1_7_8_1_–6	8	1	–6 dB
10	FR_1_7_8_5_s	8	5	Signal Only
11	FR_1_7_8_5_0	8	5	0 dB
12	FR_1_7_8_5_–6	8	5	–6 dB

Table 13 Frank Code signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 13), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. FR_1_7_4_5_s

This Frank code signal was generated with the parameters described in Table 14. The code period of the Frank-coded signal is

$$T_C = \frac{(\text{number of cycles per phase})(\text{number of phases}^2)}{\text{Carrier Frequency}} = \frac{(cpp)(N^2)}{f_c} = \frac{(5)(16)}{1000} = 80 \text{ ms.} \tag{3.16}$$

The bandwidth of the signal depends on the cycles per phase (or chirp) as

$$BW = \frac{\text{Carrier Frequency}}{\text{Cycles per phase}} = \frac{f_c}{\text{cpp}} = \frac{1000 \text{ Hz}}{5} = 200 \text{ Hz}. \tag{3.17}$$

Therefore, and according to the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB Tree to get the different output layers. Thus, Figure 43 was created which shows the output matrix from layer 6 with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. There, the

values of the carrier frequency, the bandwidth, the code period are extracted. The number of phases is extracted from Figure 44, which shows a zoom on contourplot of layer 2.

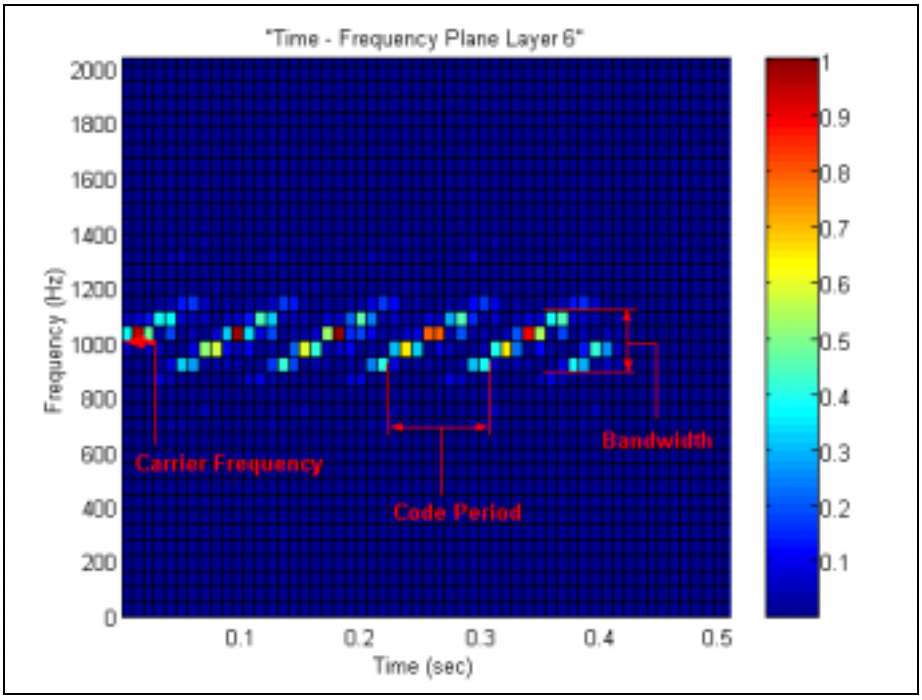

Figure 43 Output matrix from layer 6 of FR_1_7_4_5_s (colorplot).

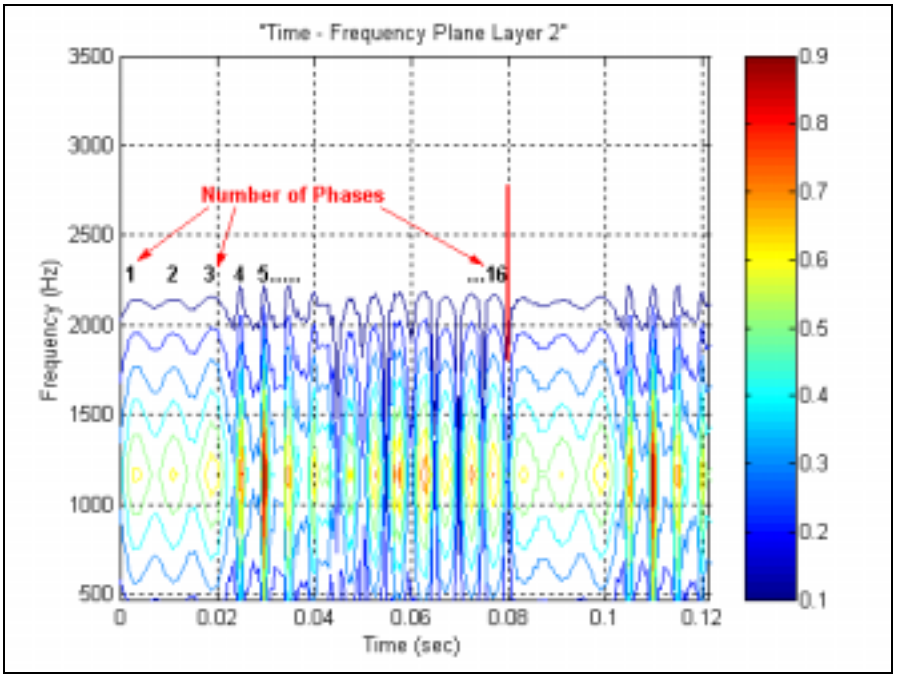

Figure 44 Zoom on output matrix from layer 2 of FR_1_7_4_5_s (contourplot).

Table 14 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

FR_1_7_4_5_s Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	16	
Cycles per phase	5	-	
SNR	Only Signal	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3.574 ms error

Table 14 Signal processing summary for FR_1_7_4_5_s.

b. FR_1_7_4_5_0

This Frank code signal was generated with the parameters described in Table 15. Figure 45 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. There the values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal.

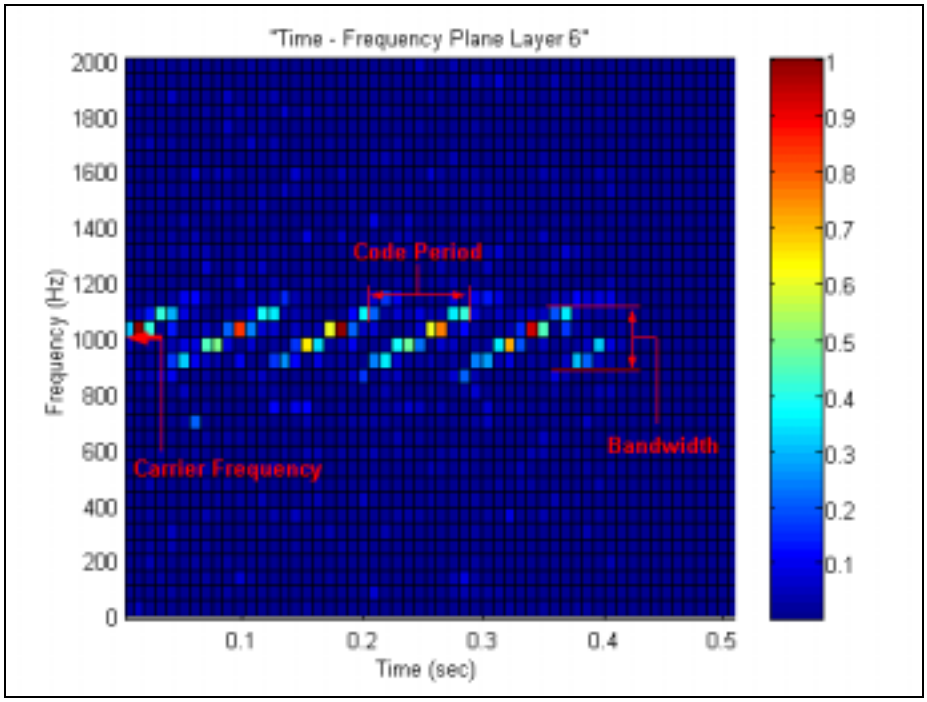

Figure 45 Output matrix from layer 6 of FR_1_7_4_5_0 (colorplot).

Table 15 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

FR_1_7_4_5_0 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	-	
Cycles per phase	5	-	
SNR	0 dB	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3.574 ms error

Table 15 Signal processing summary for FR_1_7_4_5_0.

c. FR_1_7_4_5_–6

This Frank code signal was generated with the parameters described in Table 16. Figure 46 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. There the values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal, never the less, the strong noise condition presented by the signal does not allow do it easily.

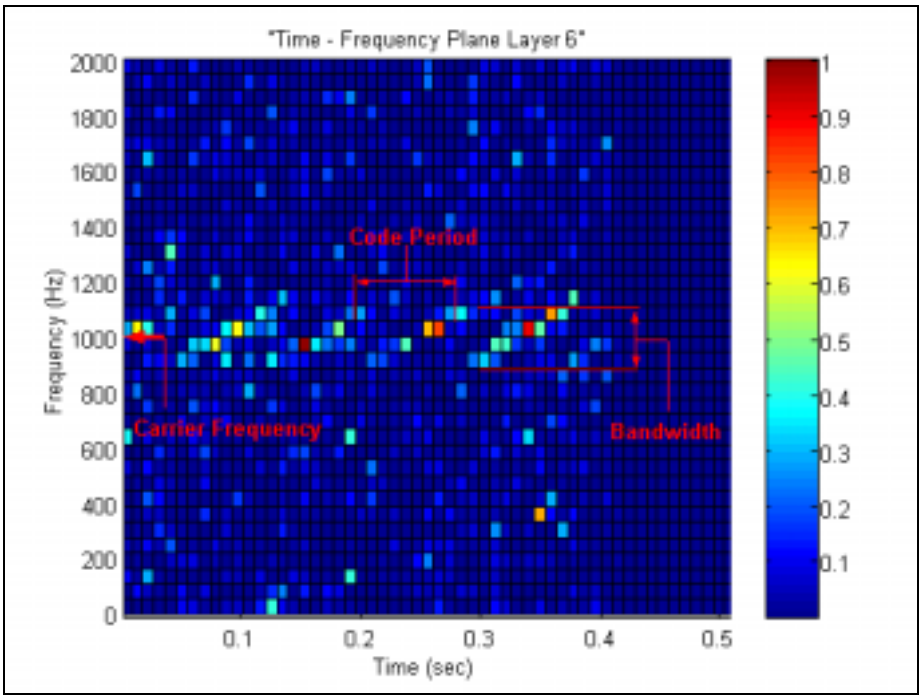

Figure 46 Output matrix from layer 6 of FR_1_7_4_5_–6 (colorplot).

Table 16 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

FR_1_7_4_5_–6 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	-	
Cycles per phase	5	-	
SNR	–6 dB	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3.574 ms error

Table 16 Signal processing summary for FR_1_7_4_5_–6.

E. P1 POLYPHASE CODE

1. Brief Description

In case of a double sideband detection (local oscillator is at band center) of a step approximation of a linear frequency modulation, a P1 code results. The P1 code also con-

sists of N^2 elements as Frank code, that way P1 code signal with $N = 4$ produces a matrix of 16 different phases, if $N = 8$ produces a matrix of 64 phases. If i is the number of sample in a given frequency and j is the number of the frequency, the phase of the i-th sample of the j-th frequency is

$$\phi_{i,j} = \frac{-\pi}{N}[N-(2j-1)][(j-1)N+(i-1)] \tag{3.18}$$

where $i = 1, 2, ..., N$, and $j = 1, 2, ..., N$.

2. Processing P1 Code Signals with QMFB Tree

The P1 code signals to be worked with are given in Table 17. All the signals were generated with a carrier frequency (f_C) of 1000 Hz and sampling frequency (f_S) of 7000 Hz. The number of phases (*N*), the number of cycles per phase (*cpp*) and the SNR are variable parameters.

	P1	**Number of code phases *N***	**Number of cycles per phase cpp**	**SNR**
1	P1_1_7_4_1_s	4	1	Signal Only
2	P1_1_7_4_1_0	4	1	0 dB
3	P1_1_7_4_1_–6	4	1	–6 dB
4	P1_1_7_4_5_s	4	5	Signal Only
5	P1_1_7_4_5_0	4	5	0 dB
6	P1_1_7_4_5_–6	4	5	–6 dB
7	P1_1_7_8_1_s	8	1	Signal Only
8	P1_1_7_8_1_0	8	1	0 dB
9	P1_1_7_8_1_–6	8	1	–6 dB
10	P1_1_7_8_5_s	8	5	Signal Only
11	P1_1_7_8_5_0	8	5	0 dB
12	P1_1_7_8_5_–6	8	5	–6 dB

Table 17 P1 signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 17), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. P1_1_7_4_5_s

This P1 code signal was generated with the parameters described in Table 18. The code period of the P1 code signal is

$$T_C = \frac{(\text{number of cycles per phase})(\text{number of phases}^2)}{\text{Carrier Frequency}} = \frac{(cpp)(N^2)}{f_c} = \frac{(5)(16)}{1000} = 80 \text{ ms.} \tag{3.19}$$

The bandwidth of the signal depends on the cycles per phase (or chirp) as

$$BW = \frac{\text{Carrier Frequency}}{\text{Cycles per phase}} = \frac{f_c}{\text{cpp}} = \frac{1000 \text{ Hz}}{5} = 200 \text{ Hz.} \tag{3.20}$$

Therefore, and according to the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB tree to obtain the different output layers. In this manner Figure 47 was created which shows the output matrix from layer 6 with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. The values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal. Figure 48 shows a zoom of the output matrix from layer 2, indicating the number of phases in the signal.

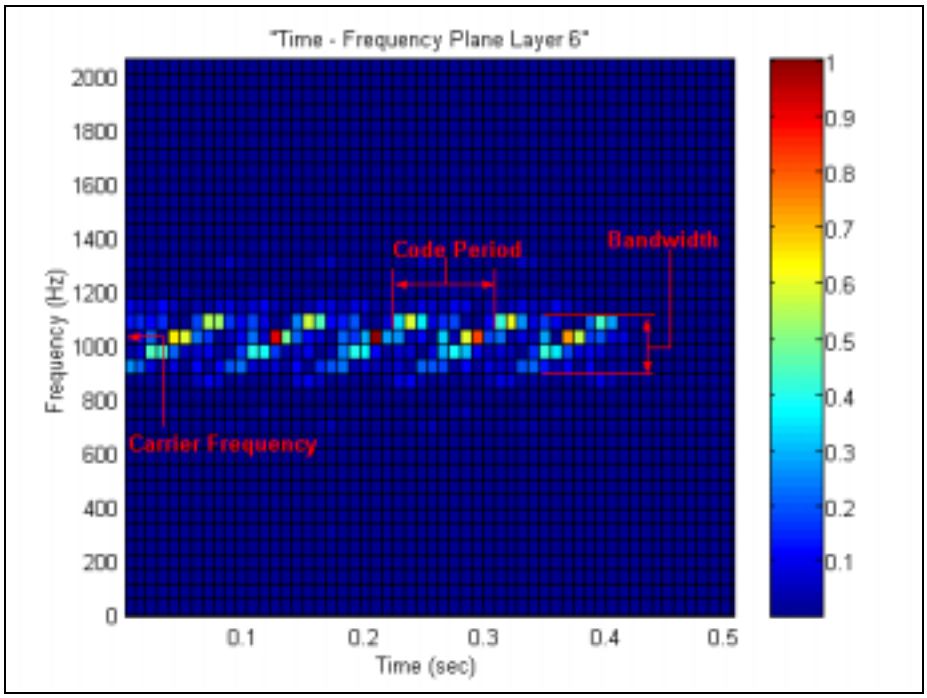

Figure 47 Output matrix from layer 7 of P1_1_7_4_5_s (colorplot).

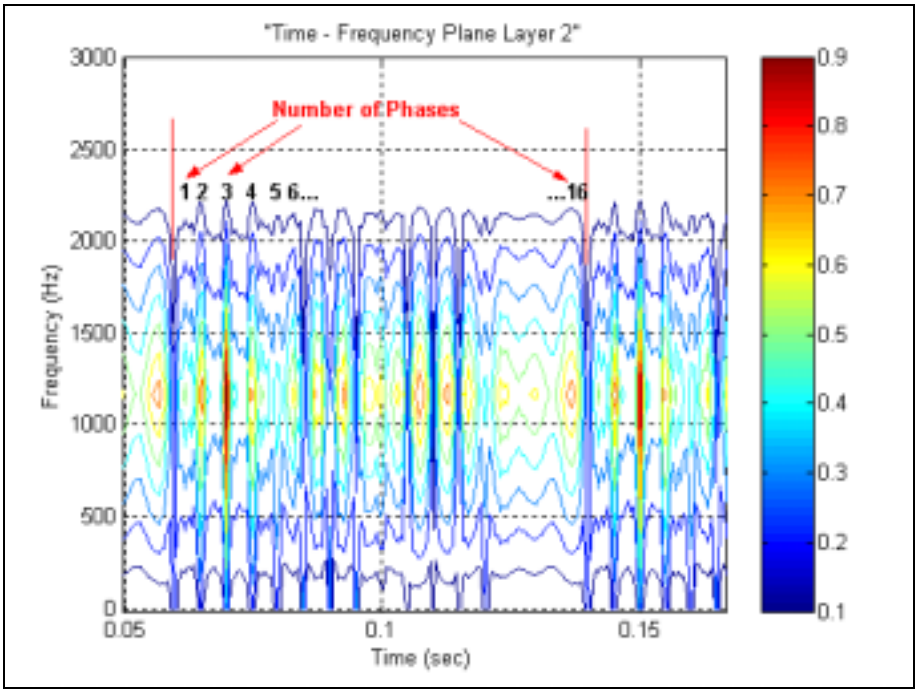

Figure 48 Zoom on output matrix from layer 2 of P1_1_7_4_5_s (contourplot).

Table 18 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P1_1_7_4_5_s Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	16	
Cycles per phase	5	-	
SNR	Signal Only	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3.574 ms error

Table 18 Signal processing summary for P1_1_7_4_5_s.

b. P1_1_7_4_5_0

This P1 code signal was generated with the parameters described in Table 19. It must be noted that this signal has almost the same parameters as the previous signal, the only difference is the SNR that now is 0 dB. Figure 49 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. There the values of the carrier frequency, the bandwidth, and the code period were extracted, giving almost the same values than the original signal.

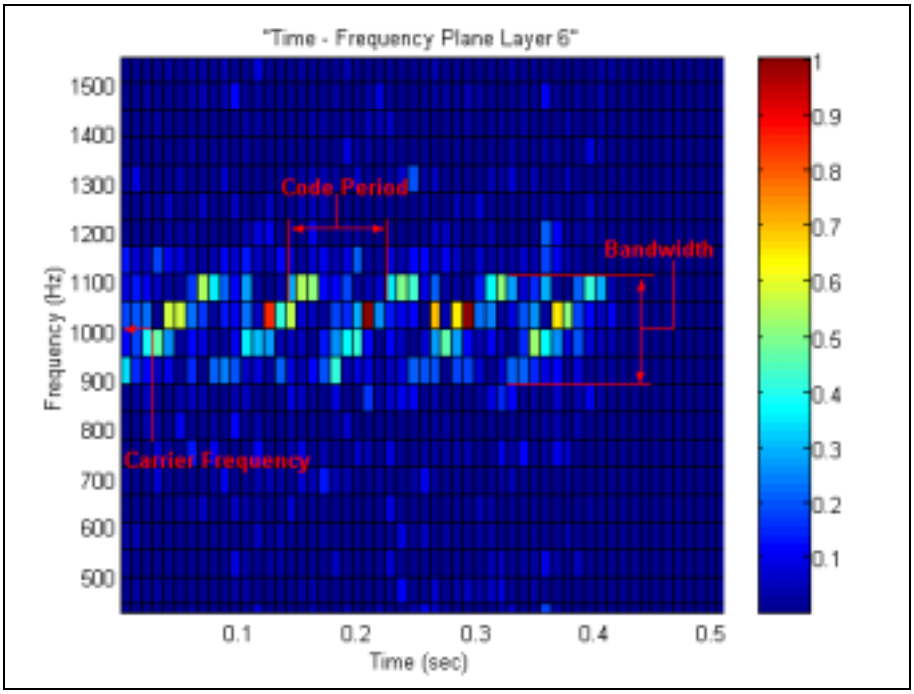

Figure 49 Output matrix from layer 7 of P1_1_7_4_5_0 (colorplot).

Table 19 shows a summary of the signal processing indicating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P1_1_7_4_5_0 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	-	
Cycles per phase	5	-	
SNR	0 dB	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3,574 ms error

Table 19 Signal processing summary for P1_1_7_4_5_0.

c. P1_1_7_4_5_–6

This P1 code signal was generated with the parameters described in Table 20. It must be noted that this signal has almost the same parameters as the previous two signals, the only difference is the signal to noise ratio that now is –6 dB. Figure 50 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. There the values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal.

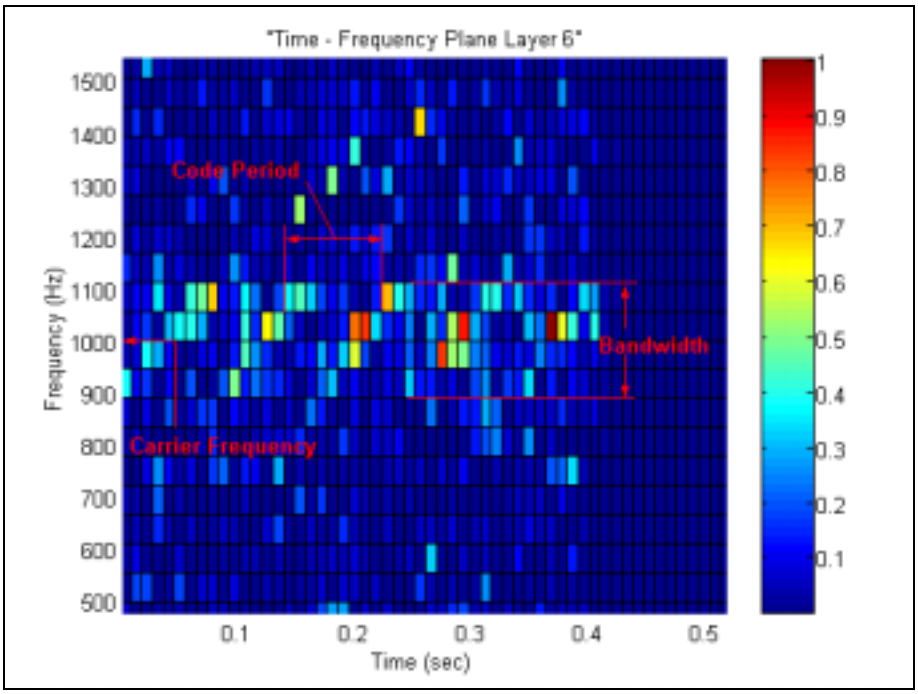

Figure 50 Output matrix from layer 7 of P1_1_7_4_5_–6 (colorplot).

Table 20 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P1_1_7_4_5_–6 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	-	
Cycles per phase	5	-	
SNR	–6 dB	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3.574 ms error

Table 20 Signal processing summary for P1_1_7_4_5_–6.

F. P2 POLYPHASE CODE

1. Brief Description

This code is essentially derived in the same way as the P1 code. The P2 code has the same phase increments within each group as the P1 code, except that the starting

phase is different. The P2 code is valid for *N* even, and each group of the code is symmetric about 0 phase. These phases can be calculated by

$$\phi_{ij} = \left\{ \frac{\pi}{2}[(N-1)/N] - [\left(\frac{\pi}{N}\right)(i-1)] \right\} [N+1-2j]. \tag{3.21}$$

This code has the frequency symmetry of the P1 code. The P2 polyphase code, as well as the P1, has more of a symmetrical spectrum than a Frank-coded signal due to its symmetry in the carrier.

2. Processing P2 Code Signals with QMFB Tree

The P2 code signals to be worked with are given in Table 21. All the signals were generated with a carrier frequency (f_C) of 1000 Hz and sampling frequency (f_S) of 7000 Hz. The number of phases (*N*), the number of cycles per phase (*cpp*), and the SNR are variable parameters.

	P2	**Number of code phases *N***	**Number of cycles per phase cpp**	**SNR**
1	P2_1_7_4_1_s	4	1	Signal Only
2	P2_1_7_4_1_0	4	1	0 dB
3	P2_1_7_4_1_–6	4	1	–6 dB
4	P2_1_7_4_5_s	4	5	Signal Only
5	P2_1_7_4_5_0	4	5	0 dB
6	P2_1_7_4_5_–6	4	5	–6 dB
7	P2_1_7_8_1_s	8	1	Signal Only
8	P2_1_7_8_1_0	8	1	0 dB
9	P2_1_7_8_1_–6	8	1	–6 dB
10	P2_1_7_8_5_s	8	5	Signal Only
11	P2_1_7_8_5_0	8	5	0 dB
12	P2_1_7_8_5_–6	8	5	–6 dB

Table 21 P2 signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 21), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. P2_1_7_4_5_s

This P2 code signal was generated with the parameters described in Table 22. The code period of the P1 code signal is

$$T_C = \frac{(\text{number of cycles per phase})(\text{number of phases}^2)}{\text{Carrier Frequency}} = \frac{(\text{cpp})(\text{N}^2)}{f_c} = \frac{(5)(16)}{1000} = 80 \text{ ms.} \tag{3.22}$$

The bandwidth of the signal depends on the cycles per phase (or chirp) as

$$BW = \frac{\text{Carrier Frequency}}{\text{Cycles per phase}} = \frac{f_c}{\text{cpp}} = \frac{1000 \text{ Hz}}{5} = 200 \text{ Hz.} \tag{3.23}$$

Now, and according with the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB Tree to get the different output layers. In this manner Figure 51 was created which show the output matrix from layer 7 with a frequency resolution of 27.777 Hz and a time resolution of 18.870 ms. The values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal. Figure 52 shows the output matrix layer 2 in a contourplot indicating the number of phases in the signal.

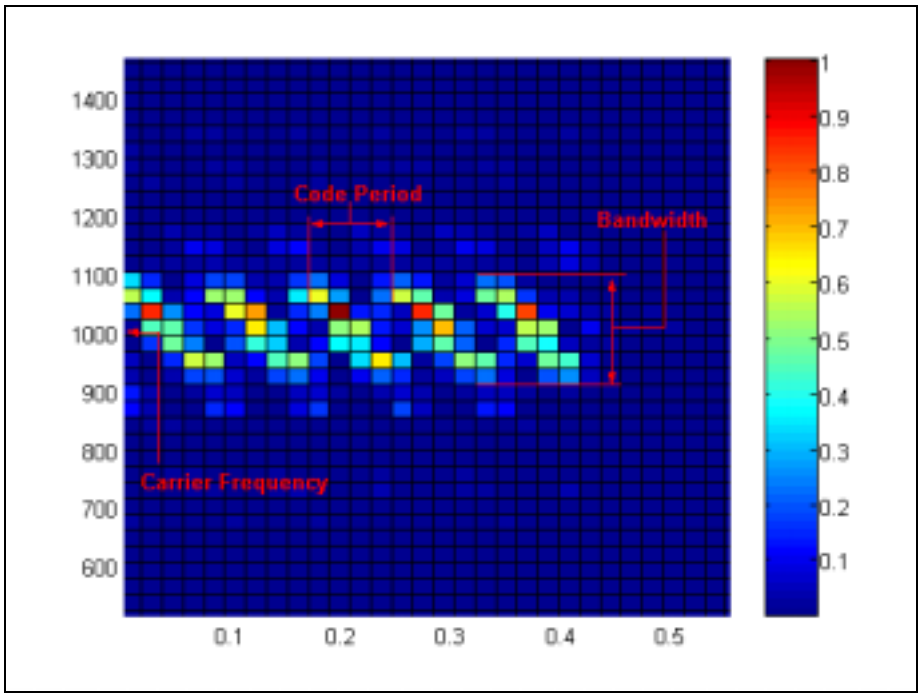

Figure 51 Output matrix from layer 7 of P2_1_7_4_5_s (colorplot).

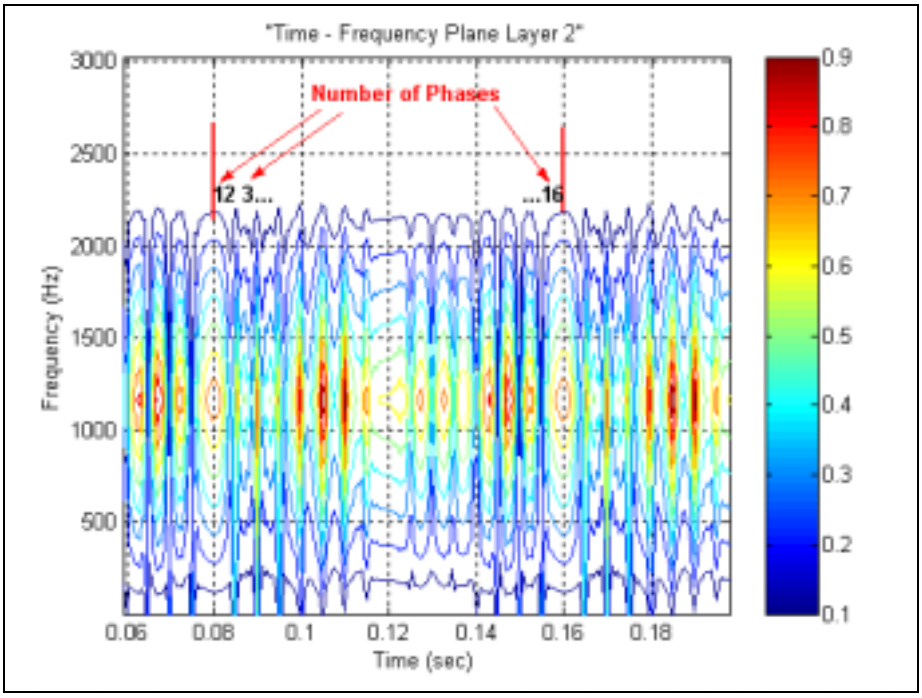

Figure 52 Zoom of output matrix from layer 2 of P2_1_7_4_5_s (contourplot).

Table 22 shows a summary of the signal processing indicating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P2_1_7_4_5_s Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	999.972 Hz	0.028 Hz error
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	16	
Cycles per phase	5	-	
SNR	Only Signal	-	
Bandwidth	200 Hz	194.439 Hz	5.561 Hz error
Code Period	80 ms	75.48 ms	4.52 ms error

Table 22 Signal processing summary for P2_1_7_4_5_s.

b. P2_1_7_4_5_0

This P2 code signal was generated with the parameters described in Table 23. It must be noted that this signal has almost the same parameters as the previous signal, the only difference is the SNR that now is 0 dB. Figure 53 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 7 when conducting the processing with the QMFB tree with a frequency resolution of 27.777 Hz and a time resolution of 18.870 ms. There the values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal.

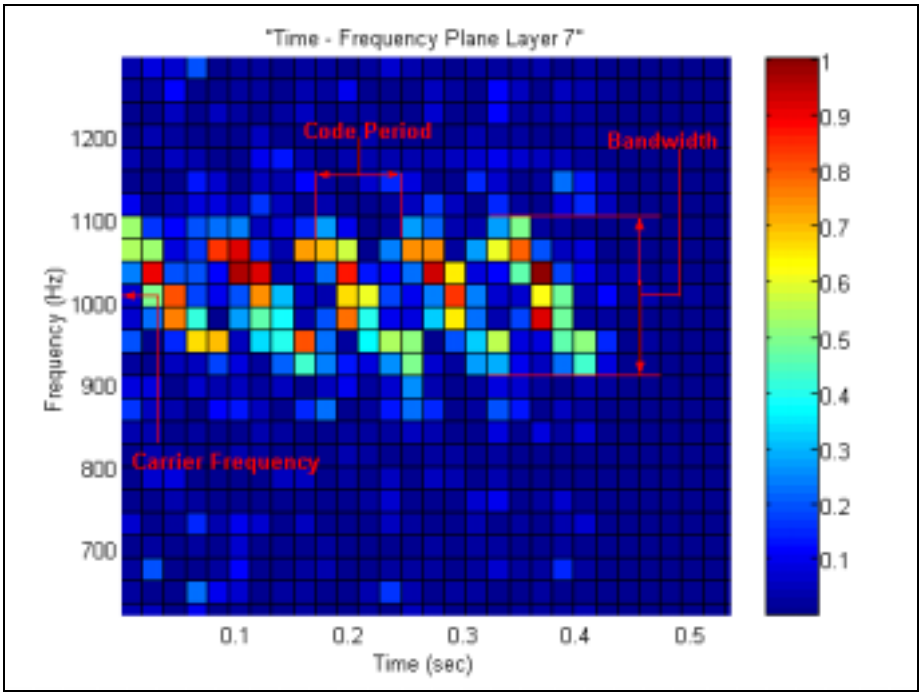

Figure 53 Output matrix from layer 7 of P2_1_7_4_5_0 (colorplot).

Table 23 shows a summary of the signal processing indicating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P2_1_7_4_5_0 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	999.972 Hz	0.028 Hz error
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	-	
Cycles per phase	5	-	
SNR	0 dB	-	
Bandwidth	200 Hz	194.439 Hz	5.561 Hz error
Code Period	80 ms	75.48 ms	4.52 ms error

Table 23 Signal processing summary for P2_1_7_4_5_0.

c. P2_1_7_4_5_–6

This P2 code signal was generated with the parameters described in Table 24. It must be noted that this signal has almost the same parameters as the previous two signals, the only difference is the signal to noise ratio that now is –6 dB. Figure 54 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 7 when conducting the processing with the QMFB tree with a frequency resolution of 27.777 Hz and a time resolution of 18.870 ms. There the values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal.

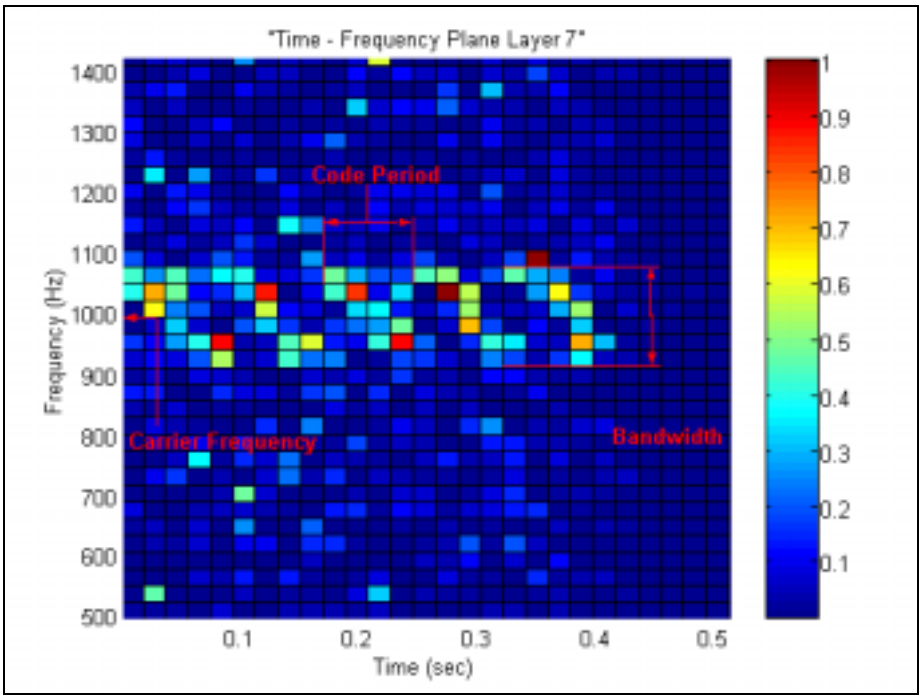

Figure 54 Output matrix from layer 7 of P2_1_7_4_5_–6 (colorplot).

Table 24 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P2_1_7_4_5_–6 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	0.999 kHz	0.028 Hz error
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	-	
Cycles per phase	5	-	
SNR	–6 dB	-	
Bandwidth	200 Hz	166.662 Hz	33.338 Hz error
Code Period	80 ms	75.48 ms	4.52 ms error

Table 24 Signal processing summary for P2_1_7_4_5_–6.

G. P3 POLYPHASE CODE

1. Brief Description

This code is derived by converting a linear-frequency modulation waveform to baseband using a local oscillator on one end of the frequency sweep and sampling the In-

phase I and Quadrature Q video at the Nyquist rate. If it is assumed that the waveform has a pulse length T in frequency $f = f_o + kt$, where k is a constant, the bandwidth of the signal will be approximately $B=kT$..

The bandwidth will support a compressed pulse length of about $t_c = 1/B$ and the waveform will provide a pulse compression ratio of $pc = T/t_c = BT$.

Assuming that the first sample of *I* and *Q* is taken at the leading edge of the waveform, the phases of successive samples taking t_c apart are:

$$\phi_i = 2\pi \int_0^{(i-1)t_c} [(f_0 + kt) - f_0]\, dt = \pi k (i-1)^2 t_c^2 \tag{3.24}$$

where $i = 1, 2, \ldots, N$. Substituting $k=B/T$ and $t_c=1/B$, the equation can be written as

$$\phi_i = \frac{\pi (i-1)^2}{BT} = \frac{\pi (i-1)^2}{N}. \tag{3.25}$$

2. Processing P3 Code Signals with QMFB Tree

The P3 code signals to be worked with are given in Table 25. All the signals were generated with a carrier frequency (f_C) of 1000 Hz and sampling frequency (f_S) of 7000 Hz. The number of phases (*N*), the number of cycles per phase (*cpp*) and the SNR are variable parameters.

	P3	Number of code phases N^2	Number of cycles per phase cpp	SNR
1	P3_1_7_16_1_s	16	1	Signal Only
2	P3_1_7_16_1_0	16	1	0 dB
3	P3_1_7_16_1_–6	16	1	–6 dB
4	P3_1_7_16_5_s	16	5	Signal Only
5	P3_1_7_16_5_0	16	5	0 dB
6	P3_1_7_16_5_–6	16	5	–6 dB
7	P3_1_7_64_1_s	64	1	Signal Only
8	P3_1_7_64_1_0	64	1	0 dB
9	P3_1_7_64_1_–6	64	1	–6 dB
10	P3_1_7_64_5_s	64	5	Signal Only
11	P3_1_7_64_5_0	64	5	0 dB
12	P3_1_7_64_5_–6	64	5	–6 dB

Table 25 P3 signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 25), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. P3_1_7_16_5_s

This P3 code signal was generated with the parameters described in Table 26. The code period of the P3 code signal is

$$T_C = \frac{(\text{number of cycles per phase})(\text{number of phases}^2)}{\text{Carrier Frequency}} = \frac{(\text{cpp})(\text{N}^2)}{f_c} = \frac{(5)(16)}{1000} = 80 \text{ ms.} \tag{3.26}$$

The bandwidth of the signal depends on the cycles per phase (or chirp) as

$$BW = \frac{\text{Carrier Frequency}}{\text{Cycles per phase}} = \frac{f_c}{\text{cpp}} = \frac{1000 \text{ Hz}}{5} = 200 \text{ Hz.} \tag{3.27}$$

Now, and according with the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB tree to get the different output layers. In this manner, Figure 55 was created which shows the output matrix from layer 6 with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. The

values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal. Figure 56 shows a zoom of the output matrix layer 2 indicating the number of phases in the signal.

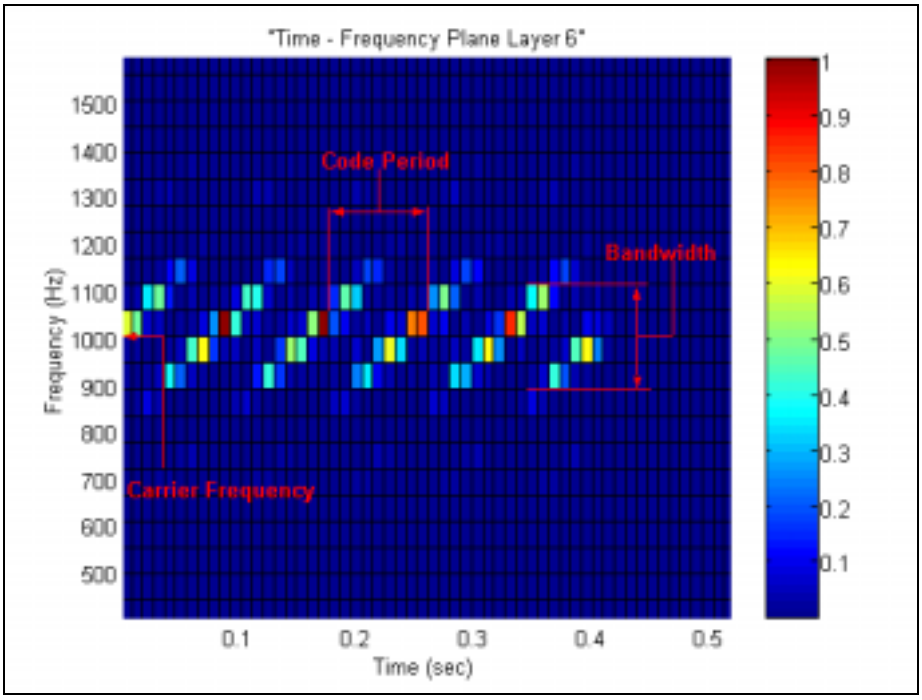

Figure 55 Output matrix from layer 7 of P3_1_7_16_5_s (colorplot).

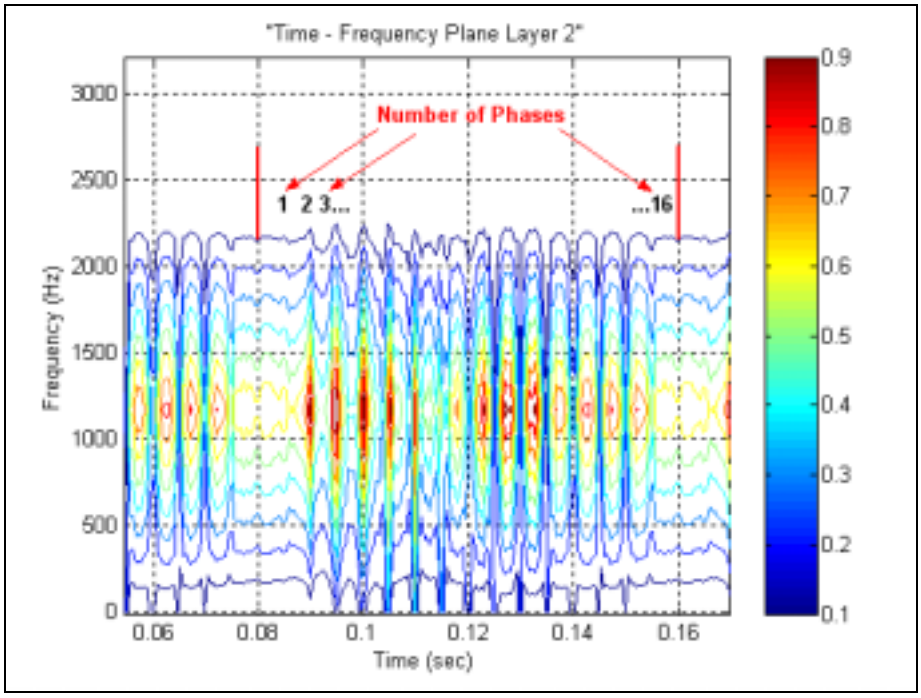

Figure 56 Zoom of output matrix layer 2 of P3_1_7_16_5_s (contourplot).

Table 26 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P3_1_7_16_5_s Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	16	
Cycles per phase	5	-	
SNR	Only Signal	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3.574 ms error

Table 26 Signal processing summary for P3_1_7_16_5_s.

b. P3_1_7_16_5_0

This P3 code signal was generated with the parameters described in Table 27. It must be noted that this signal has almost the same parameters than the previous signal, the only difference is the signal to noise ratio that now is 0 dB. Figure 57 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. The values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal.

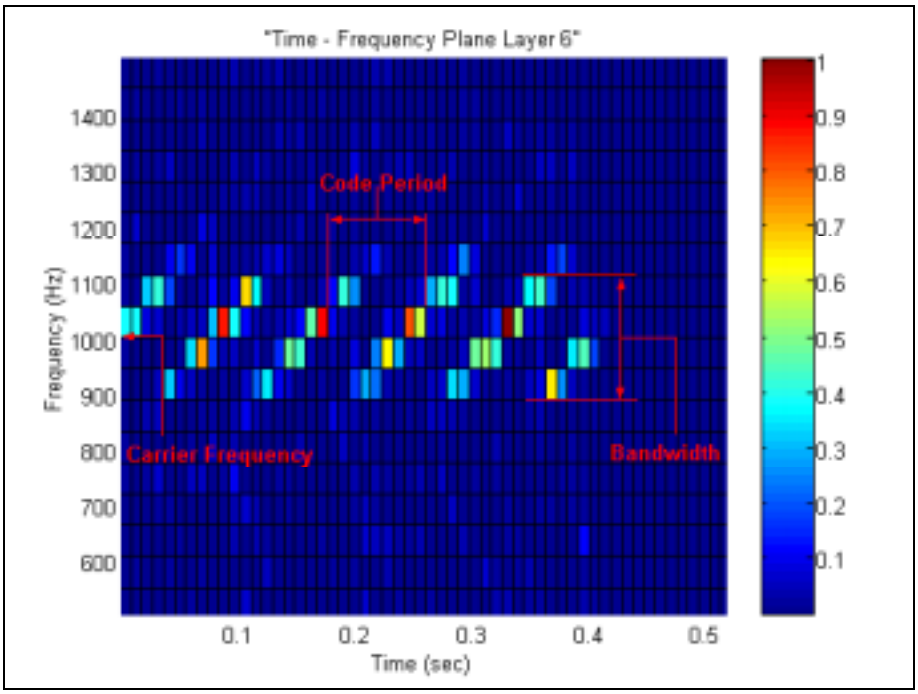

Figure 57 Output matrix from layer 7 of P3_1_7_16_5_0 (colorplot).

Table 27 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P3_1_7_16_5_0 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	-	
Cycles per phase	5	-	
SNR	0 dB	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	92.86 ms	12.86 ms error

Table 27 Signal processing summary for P3_1_7_16_5_0.

c. P3_1_7_16_5_–6

This P3 code signal was generated with the parameters described in Table 28. It must be noted that this signal has almost the same parameters as the previous two signals, the only difference is the signal to noise ratio that now is –6 dB. Figure 58 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a

frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. There the values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal.

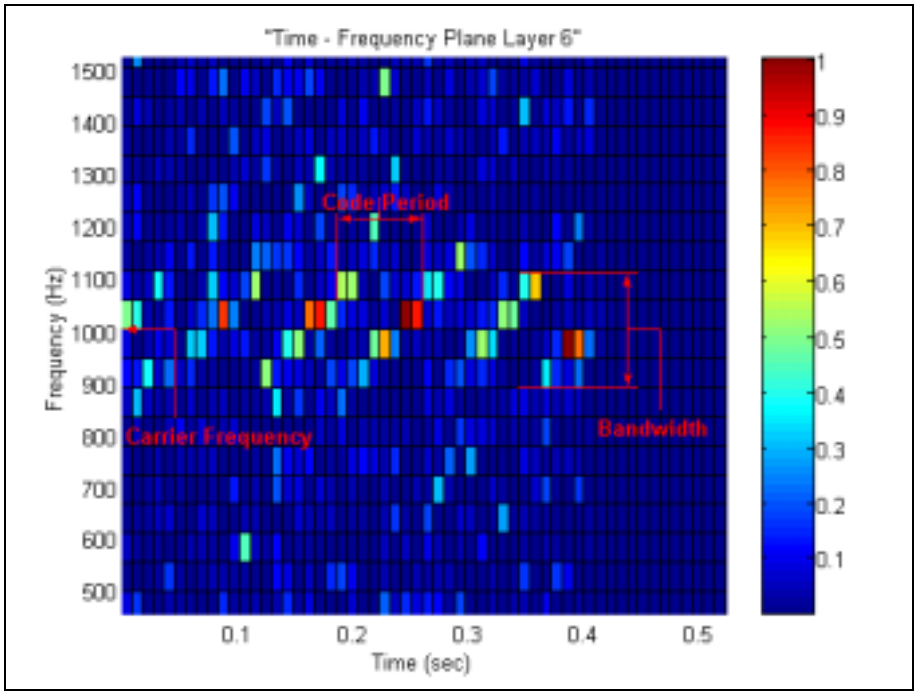

Figure 58 Output matrix from layer 7 of P3_1_7_16_5_–6 (colorplot).

Table 28 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P3_1_7_16_5_–6 Parameters	**Generation**	**Detection**	**Comment**
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	-	
Cycles per phase	5	-	
SNR	–6 dB	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	92.86 ms	12.86 ms error

Table 28 Signal processing summary for P3_1_7_16_5_–6.

H. P4 POLYPHASE CODE

1. Brief Description

Conceptual coherent double sideband detection of the linear frequency modulation waveform and sampling at the Nyquist rate yields a polyphase code named the P4. The P4 code consists of discrete phases of the linear chirp waveform taken at specific time intervals and exhibits the same range Doppler coupling associate with the chirp waveform. However, the peak sidelobe levels are lower than those of the unweighted chirp waveform. Various weighting techniques can be applied to reduce the sidelobe levels further. Phase code elements of the P4 code are given by [1]

$$\phi_i = \frac{\pi(i-1)^2}{N} - \pi(i-1) \tag{3.28}$$

for $i = 1$ to N.

2. Processing P4 Code Signals with QMFB Tree

The P4 code signals to be worked with are given in Table 29. All the signals were generated with a carrier frequency (f_C) of 1000 Hz and sampling frequency (f_S) of 7000 Hz. The number of phases (N), the number of cycles per phase (cpp) and the SNR are variable parameters.

	P4	Number of code phases N^2	Number of cycles per phase cpp	SNR
1	P4_1_7_16_1_s	16	1	Signal Only
2	P4_1_7_16_1_0	16	1	0 dB
3	P4_1_7_16_1_–6	16	1	–6 dB
4	P4_1_7_16_5_s	16	5	Signal Only
5	P4_1_7_16_5_0	16	5	0 dB
6	P4_1_7_16_5_–6	16	5	–6 dB
7	P4_1_7_64_1_s	64	1	Signal Only
8	P4_1_7_64_1_0	64	1	0 dB
9	P4_1_7_64_1_–6	64	1	–6 dB
10	P4_1_7_64_5_s	64	5	Signal Only
11	P4_1_7_64_5_0	64	5	0 dB
12	P4_1_7_64_5_–6	64	5	–6 dB

Table 29 P4 signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 29), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. P4_1_7_16_5_s

This P4 code signal was generated with the parameters described in Table 30. The code period of the P4 code signal is

$$T_C = \frac{(\text{number of cycles per phase})(\text{number of phases}^2)}{\text{Carrier Frequency}} = \frac{(\text{cpp})(\text{N}^2)}{f_c} = \frac{(5)(16)}{1000} = 80 \text{ ms.} \tag{3.29}$$

The bandwidth of the signal depends on the cycles per phase (or chirp) as

$$BW = \frac{\text{Carrier Frequency}}{\text{Cycles per phase}} = \frac{f_c}{\text{cpp}} = \frac{1000 \text{ Hz}}{5} = 200 \text{ Hz.} \tag{3.30}$$

Now, and according with the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB Tree to obtain the different output layers. In this manner, Figure 59 was created which show the output matrix from layer 6 with a frequency resolution of 55.5556 Hz and a time resolution of 9.286

ms. The values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal. Figure 60 shows a zoom of output matrix layer 2 indicating the number of phases in the signal.

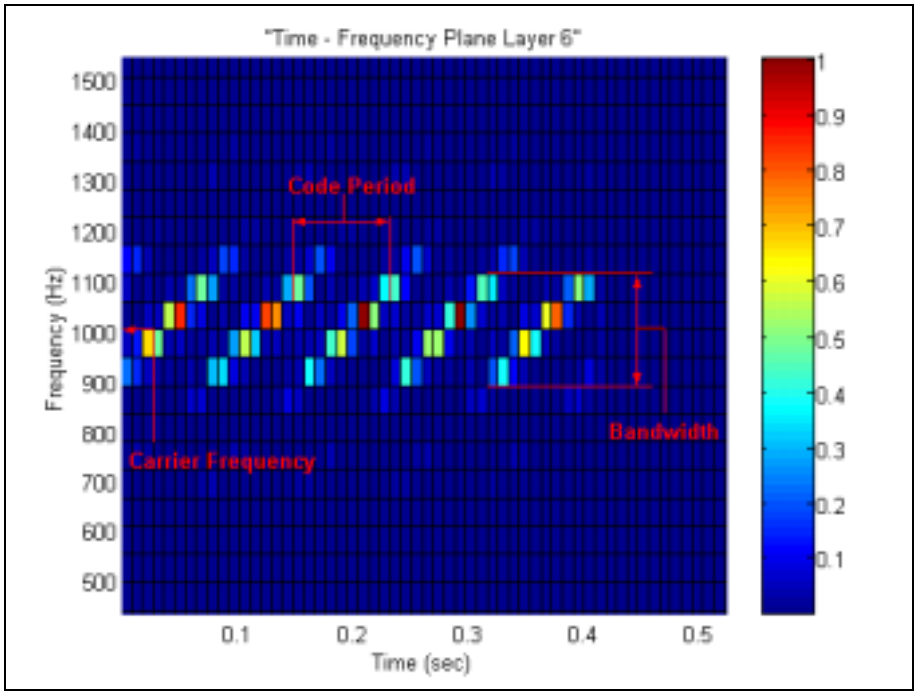

Figure 59 Output matrix from layer 7 of P4_1_7_16_5_s (colorplot).

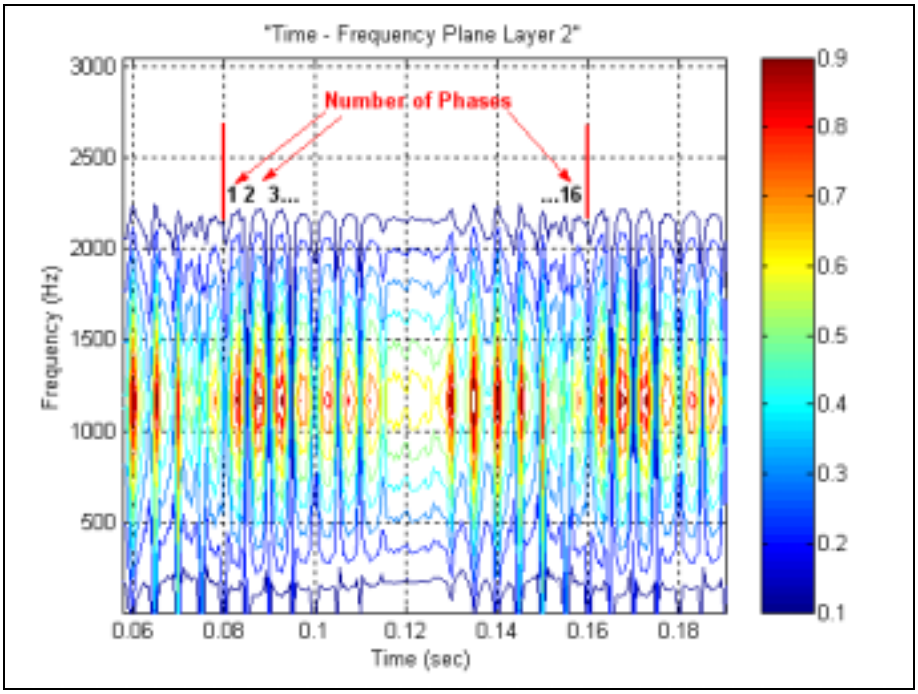

Figure 60 Zoom of output matrix layer 2 of P4_1_7_16_5_s (colorplot).

Table 30 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P4_1_7_16_5_s Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	16	
Cycles per phase	5	-	
SNR	Only Signal	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3.574 ms error

Table 30 Signal processing summary for P4_1_7_16_5_s.

b. P4_1_7_16_5_0

This P4 code signal was generated with the parameters described in Table 31. It must be noted that this signal has almost the same parameters as the previous signal, the only difference is the SNR that now is 0 dB. Figure 61 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. There the values of the carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal.

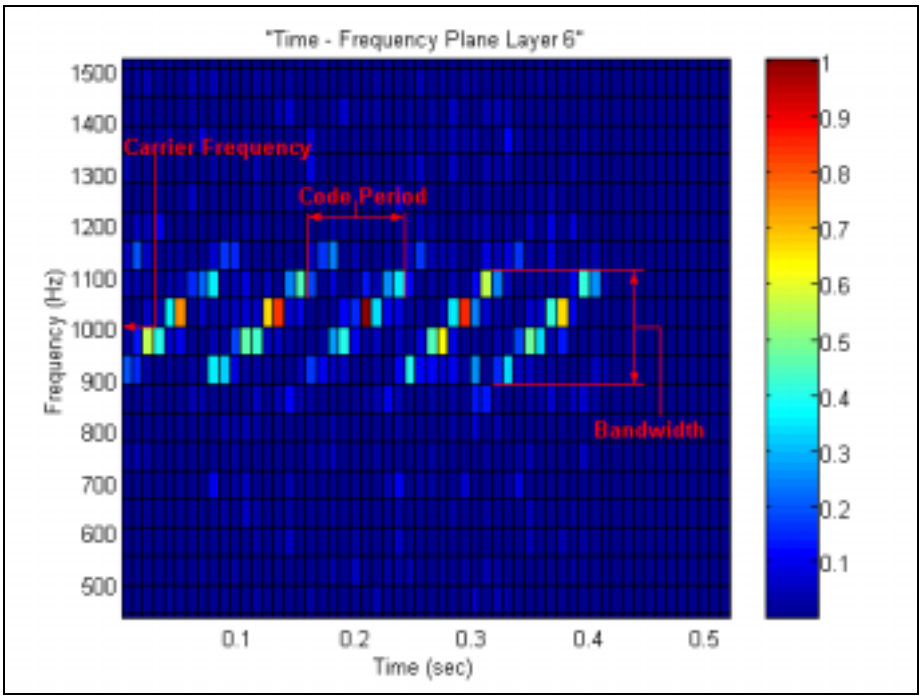

Figure 61 Output matrix from layer 7 of P4_1_7_16_5_0 (colorplot).

Table 31 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P4_1_7_16_5_0 Parameters	Generation	Detection	Comment
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	16	
Cycles per phase	5	-	
SNR	0 dB	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3.574 ms error

Table 31 Signal processing summary for P4_1_7_16_5_0.

c. P4_1_7_16_5_–6

This P4 code signal was generated with the parameters described in Table 32. It must be noted that this signal has almost the same parameters as the previous two signals, the only difference is the signal to noise ratio that now is –6 dB. Figure 62 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 55.5556 Hz and a time resolution of 9.286 ms. The values of the

carrier frequency, the bandwidth, and the code period are extracted, giving almost the same values than the original signal.

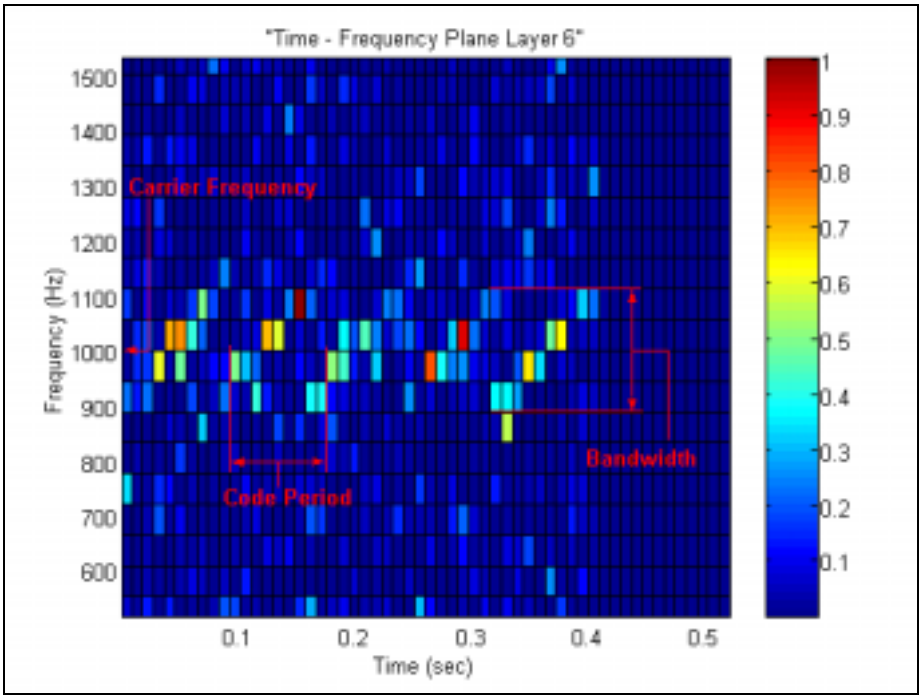

Figure 62 Output matrix from layer 7 of P3_1_7_16_5_–6 (colorplot).

Table 32 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

P4_1_7_16_5_–6 Parameters	**Generation**	**Detection**	**Comment**
Carrier Frequency	1 kHz	1 kHz	
Sampling Frequency	7 kHz	7 kHz	Given
Number of Phases	16	-	
Cycles per phase	5	-	
SNR	–6 dB	-	
Bandwidth	200 Hz	222.222 Hz	22.222 Hz error
Code Period	80 ms	83.574 ms	3.574 ms error

Table 32 Signal processing summary for P3_1_7_16_5_–6.

I. COSTAS CODE

1. Brief Description

In a frequency hopping system, the signal consists of one or more frequencies being chosen from a set $\{f1, f2, ..., fm\}$ of available frequencies, for transmission at each of a set $\{t1, t2, ..., tn\}$ of consecutives time intervals. For modeling purposes, it is reasonable to consider the situation in which $m = n$, and a different one of n equally spaced frequencies $\{f1, f2, ..., fn\}$ is transmitted during each of the n equal duration time intervals $\{t1, t2, ..., tn\}$. Such a signal is represented by a $n \times n$ permutation matrix *A*, where the n rows correspond to the n frequencies, the n columns correspond to the n intervals, and the entry a_{ij} equals 1 means transmission and 0 otherwise [7].

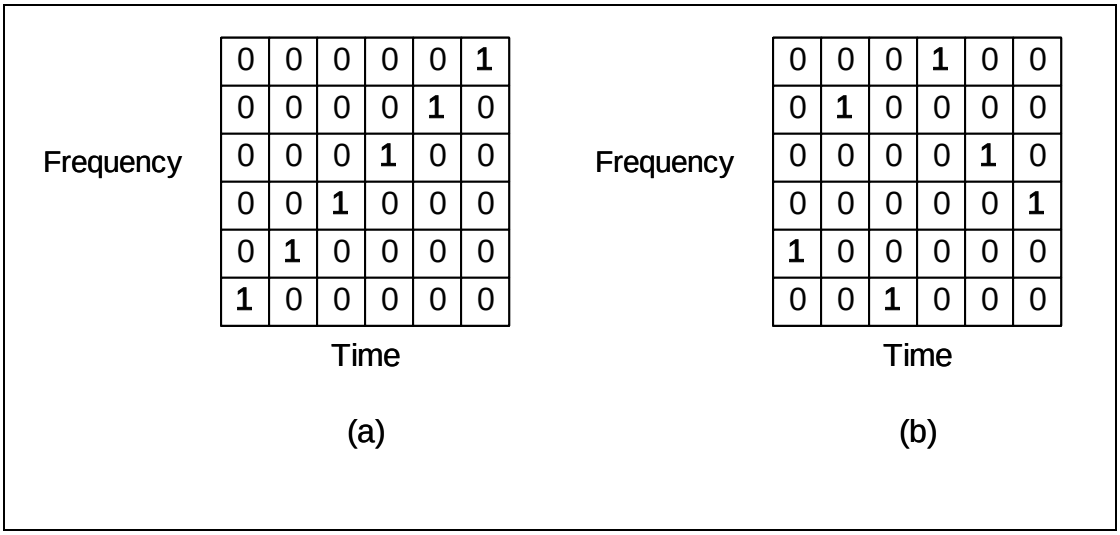

Figure 63 Binary matrix representation of (a) quantized linear FM and (b) Costas Signal.

This signifies that, at any given time, a slotone frequency is transmitted, and each frequency is transmitted only one (Figure 63(a)). Other possible frequency-hopping sequences that belong to this family. This hopping order stongly effects the ambiguity function of these signals. Frequency-hopping signals allow a simple procedure that results in a rough approximation of their ambiguity function. This is possible because the cross correlation signals at different frequencies approaches zero when the frequence

difference is large relative to the inverse of the signal duration. The ambiguity function, at any given coordinates, is an integral of the product between the original signal and a replica of it, which is shifted in time and frequency according to the delay and the Doppler coordinates of the function.

Performing an exercise on the matrix in Figure 63(b), results show that except for the zero-shift cases, when the number of coincidences is *N*, finding a combination of shifts yielding more than one coincidence is not possible. This is actually the criteria of Costas sequences, where sequences of frequency hopping yield no more than one coincidence. For example: if $\{a_j\} = 4,7,1,6,5,2,3$ is a Costas sequence, then its coding matrix and difference matrix are shown in Figure 64.

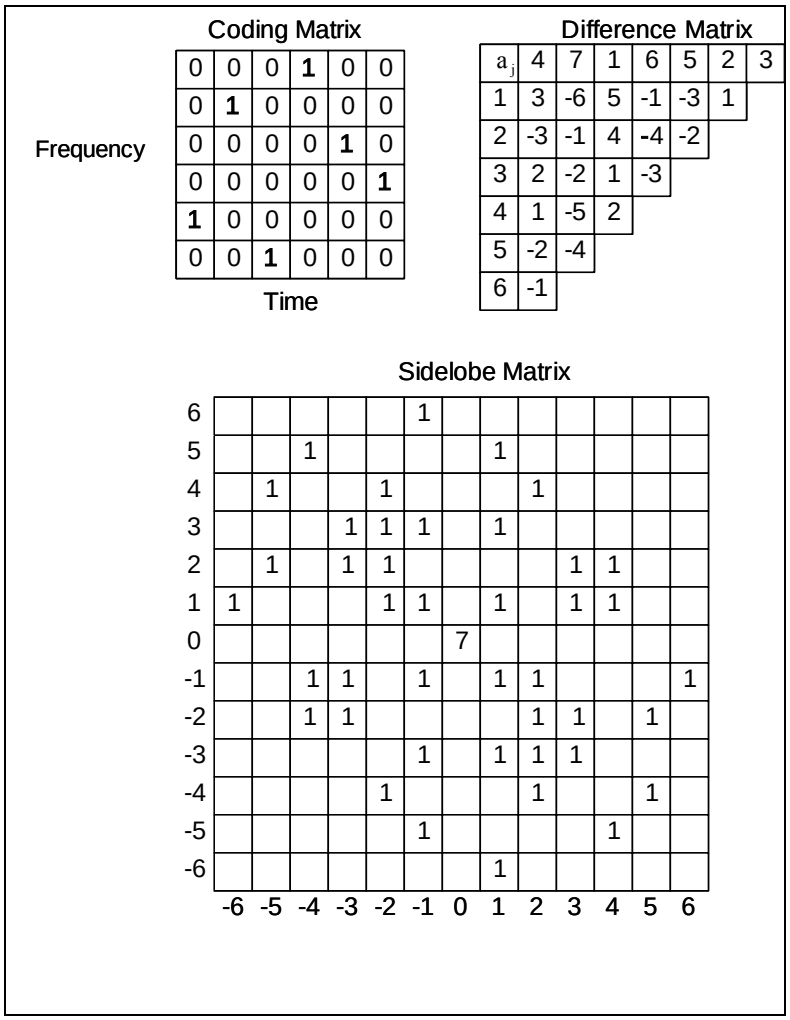

Figure 64 The coding matrix, different matrix and ambiguity sidelobes matrix of a Costas signal.

2. Processing Costas Code Signals with QMFB Tree

The Costas code signals to be worked with are given in Table 33. All the signals were generated with a carrier frequency (f_C) of 1000 or 3000 Hz, a sampling frequency (f_S) of 15000 Hz, and the frequency sequence 4-7-1-6-5-2-3 (each of this numbers represents frequencies in kHz), with transmission time in each frequency equal to 10 or 20 ms. The number of cycles per phase (*cpp*) and the SNR are variable parameters.

	COSTAS	Sequence	Number of cycles per phase cpp	SNR
1	C_1_15_10_s	4716523	10	Signal Only
2	C_1_15_10_0	4716523	10	0 dB
3	C_1_15_10_–6	4716523	10	–6 dB
4	C_1_15_20_s	4716523	20	Signal Only
5	C_1_15_20_0	4716523	20	0 dB
6	C_1_15_20_–6	4716523	20	–6 dB
7	C_2_17_10_s	2638751	10	Signal Only
8	C_2_17_10_0	2638751	10	0 dB
9	C_2_17_10_–6	2638751	10	–6 dB
10	C_2_17_20_s	2638751	20	Signal Only
11	C_2_17_20_0	2638751	20	0 dB
12	C_2_17_20_–6	2638751	20	–6 dB

Table 33 Costas signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 33), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. C_1_15_10_s

This Costas code signal was generated with the parameters described in Table 34. The frequency sequence is the sequence number 1 (4-7-1-6-5-2-3 kHz) with 10 cycles per phase. The transmission time in each frequency of the sequence is equal to 10 ms. Now, and according with the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB Tree to obtain the different output layers. In this manner Figure 65 was created which show the output matrix from layer 6

with a frequency resolution of 124.754 Hz. A zoom of a previous layer output matrix is shown in Figure 66; the values of the Costas frequency sequence are identified, as well as the code period and bandwidth, giving the same values than the original signal.

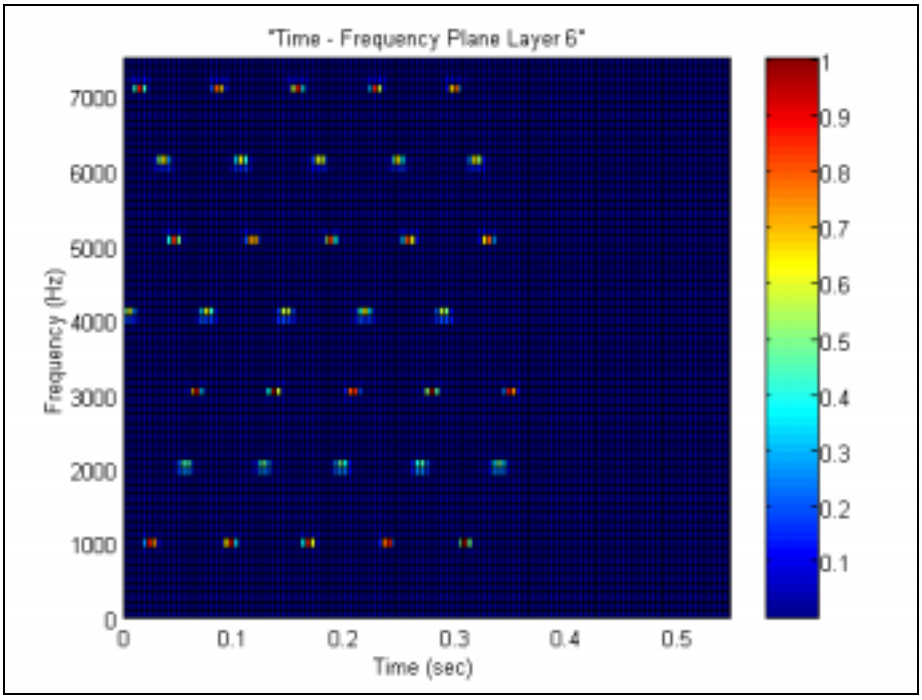

Figure 65 Output matrix from layer 6 of C_1_15_10_s (colorplot).

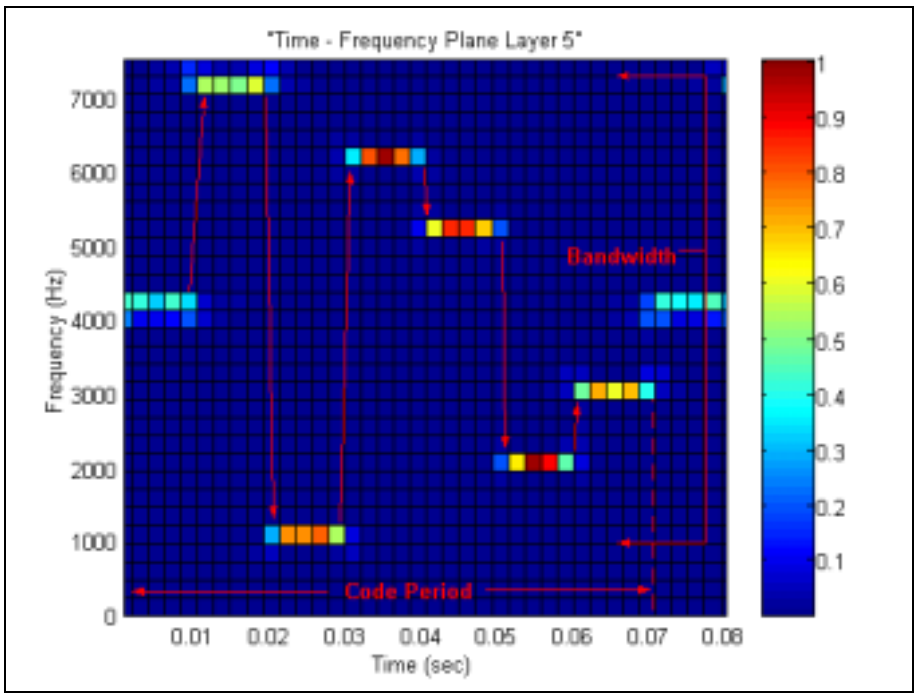

Figure 66 Zoom of output matrix from layer 5 of C_1_15_10_s (colorplot).

Table 34 presents a summary of the signal processing indicating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

C_1_15_10_s Parameters	Generation	Detection	Comment
Costas Sequence	4-7-1-6-5-2-3 kHz	4-7-1-6-5-2-3 kHz	
Sampling Frequency	15 kHz	15 kHz	Given
SNR	Only Signal	-	
Transmission time per Frequency	10 ms	10 ms	
Code Period	70 ms	70 ms	
Bandwidth	6000 Hz	6112.978 Hz	112.978 Hz error

Table 34 Signal processing summary for C_1_15_10_s.

b. C_1_15_10_0

This Costas code signal was generated with the parameters described in Table 35. It must be noted that this signal has almost the same parameters as the previous, the only difference is the SNR that now is 0 dB. Figure 67 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 124.754 Hz. A zoom of a previous layer output matrix is shown in Figure 68; the values of the Costas frequency sequence are identified, as well as the code period and bandwidth, giving almost the same values than the original signal.

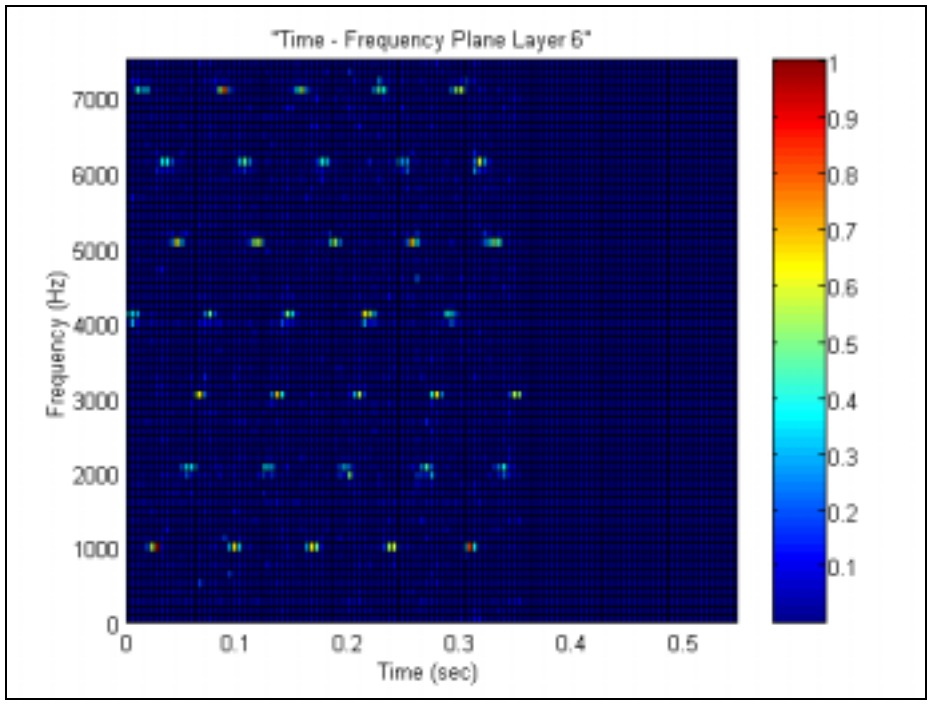

Figure 67 Output matrix from layer 7 of C_1_15_10_0 (colorplot).

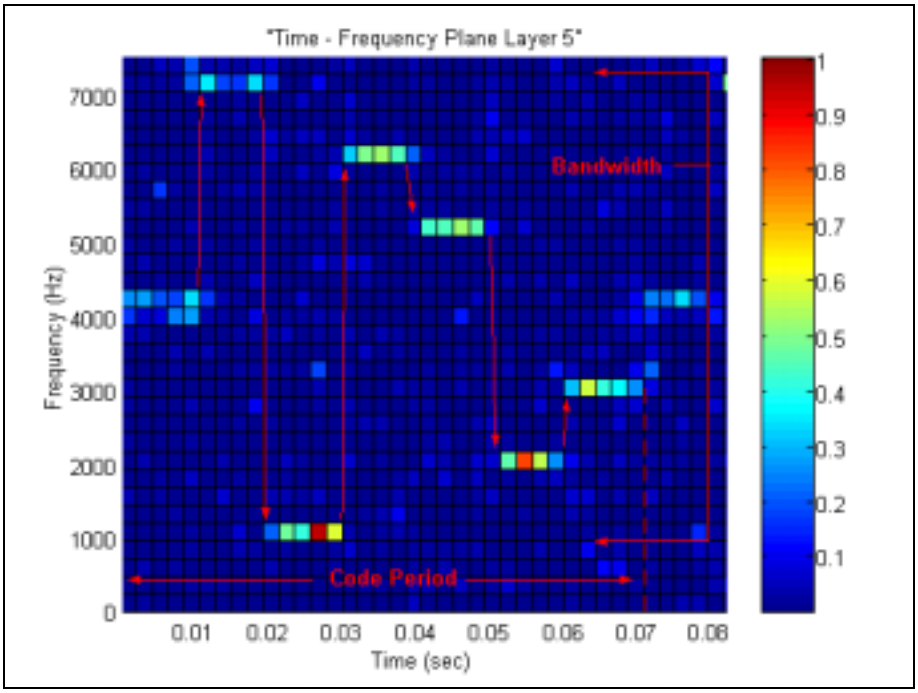

Figure 68 Zoom of output matrix from layer 5 of C_1_15_10_0 (colorplot).

Table 35 presents a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

C_1_15_10_0 Parameters	Generation	Detection	Comment
Costas Sequence	4-7-1-6-5-2-3 kHz	4-7-1-6-5-2-3 kHz	
Sampling Frequency	15 kHz	15 kHz	Given
SNR	0 dB	-	
Transmission time per Frequency	10 ms	10 ms	
Code Period	70 ms	70 ms	
Bandwidth	6000 Hz	6112.978 Hz	112.978 Hz error

Table 35 Signal processing summary for C_1_15_10_0.

c. C_1_15_10_–6

This Costas code signal was generated with the parameters described in Table 36. It must be noted that this signal has almost the same parameters as the previous two signals, the only difference is the signal to noise ratio that now is –6 dB. Figure 69 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 6 when conducting the processing with the QMFB tree with a frequency resolution of 124.754 Hz. A zoom of a previous layer output matrix is shown in Figure 70; the values of the Costas frequency sequence are hardly identified, as well as the code period and bandwidth, giving roughly the same values than the original signal.

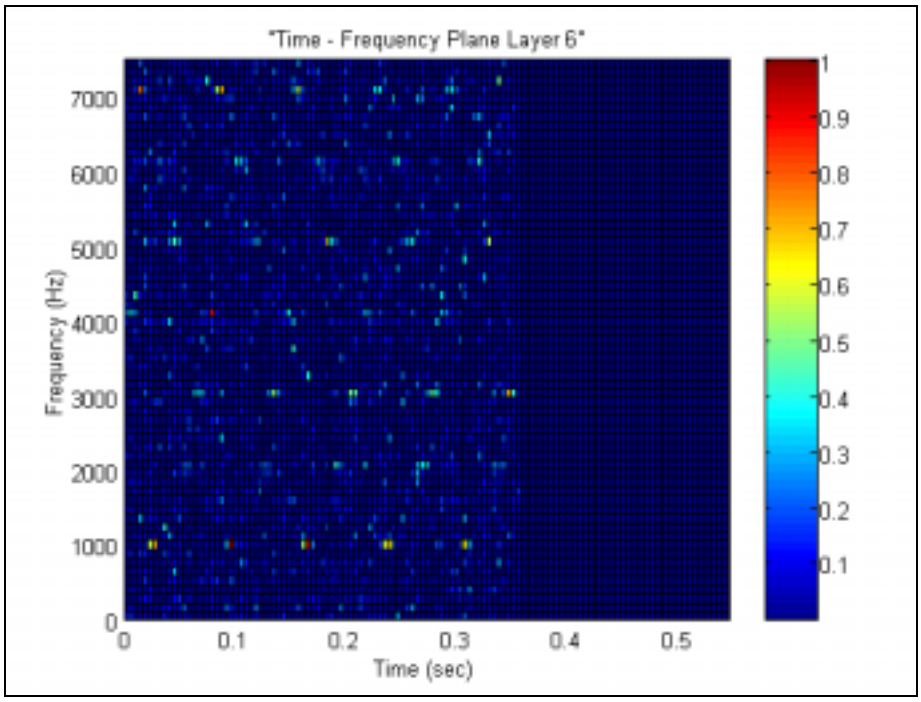

Figure 69 Output matrix from layer 7 of C_1_15_10_–6 (colorplot).

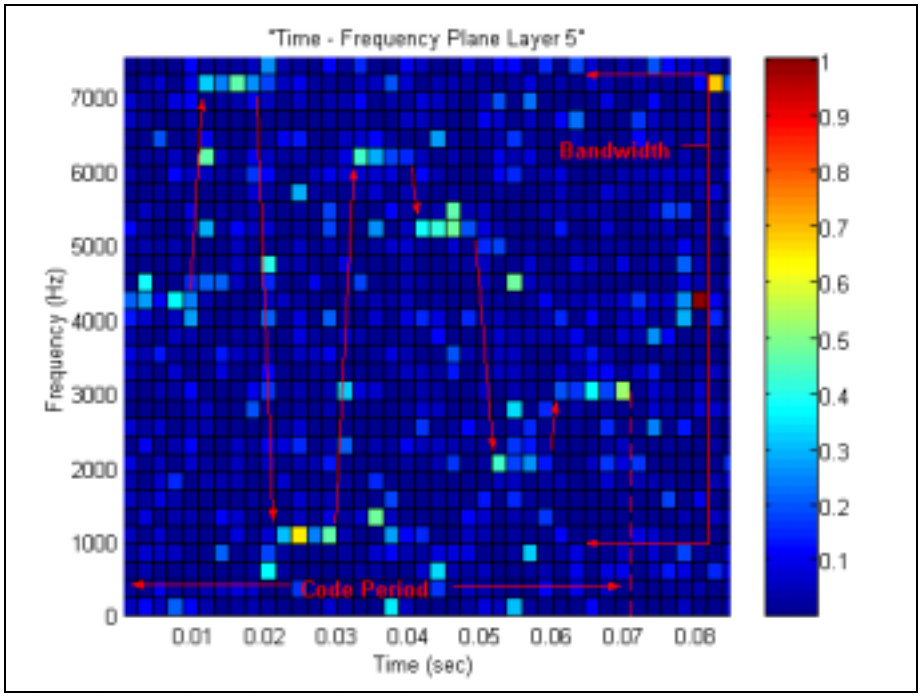

Figure 70 Zoom of output matrix from layer 5 of C_1_15_10_–6 (colorplot).

Table 36 shows a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

C_1_15_10_–6 Parameters	Generation	Detection	Comment
Costas Sequence	4-7-1-6-5-2-3 kHz	4-7-1-6-5-2-3 kHz	
Sampling Frequency	15 kHz	15 kHz	Given
SNR	–6 dB	-	
Transmission time per Frequency	10 ms	10 ms	
Code Period	70 ms	70 ms	
Bandwidth	6000 Hz	6112.978 Hz	112.978 Hz error

Table 36 Signal processing summary for C_1_15_10_–6.

J. FREQUENCY SHIFT KEYING/PHASE SHIFT KEYING COMBINED WITH COSTAS CODE (FSK/PSK COSTAS)

1. Brief Description

This modulation technique is the result of a combination of frequency shift keying based on a Costas frequency hopping matrix and phase shift keying using Barker sequences of different lengths. In a Costas frequency hopped signal, the firing order defines what frequencies will appear and with what duration. During each sub-period, as the signal stays in one of the frequencies, a binary phase modulation occurs according to a Barker sequence of length five, seven, eleven or thirteen bits.

The final waveform may be seen as a binary phase shifting modulation within each frequency hop, resulting in 25 phase slots equally distributed in each frequency. As illustrated in Figure 71, if we consider N_F as the number of frequency hops and N_P as the number of phase slots of duration T_P in each sub-period T_F, the total number of phase slots in the FSK/PSK waveform is given by:

$$N = N_F N_P. \tag{3.31}$$

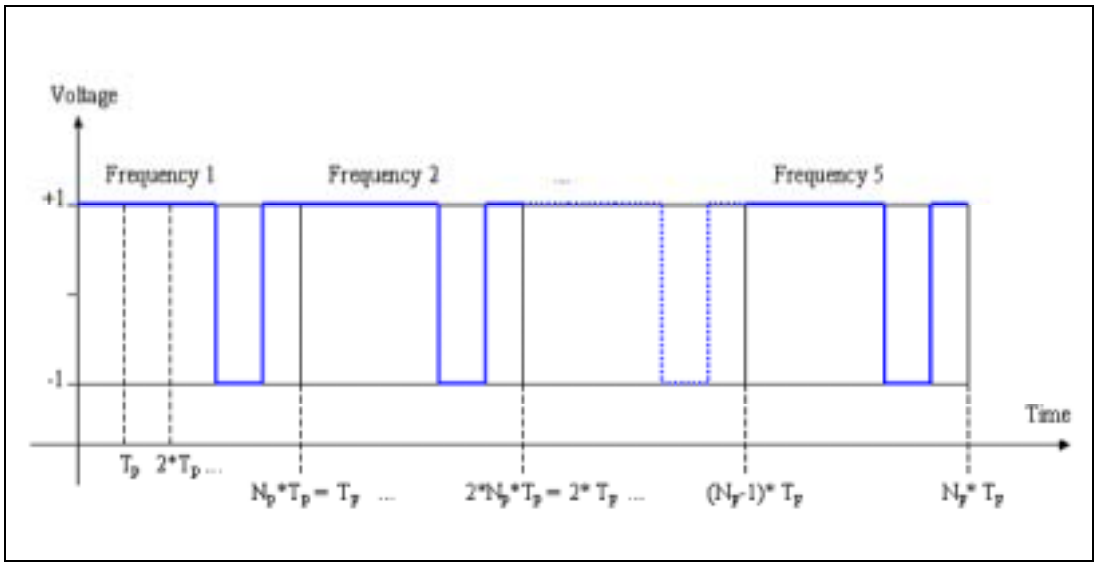

Figure 71 General FSK/PSK Signal containing N_F frequency hops with N_P phase slots per frequency.

2. Processing FSK/PSK Costas Code Signals with QMFB Tree

The FSK/PSK Costas code signals to be worked with are given in Table 33. All the signals were generated with a Costas sequence of 7 frequencies (4-7-1-6-5-2-3 kHz), sampling frequency equal to 15000 Hz, the number of bits Barker code, the cycles per bit, and the SNR are variables parameters.

	FSK/PSK Costas	Sequence	Barker Bits	Number of cycles per bit cpp	SNR
1	FSK_PSK_C_1_15_5_1_s	4716523	5	1	Signal Only
2	FSK_PSK_C_1_15_5_1_0	4716523	5	1	0 dB
3	FSK_PSK_C_1_15_5_5_s	4716523	5	5	Signal Only
4	FSK_PSK_C_1_15_5_5_0	4716523	5	5	0 dB
5	FSK_PSK_C_1_15_11_1_s	4716523	11	1	Signal Only
6	FSK_PSK_C_1_15_11_1_0	4716523	11	1	0 dB
7	FSK_PSK_C_1_15_11_5_s	4716523	11	5	Signal Only
8	FSK_PSK_C_1_15_11_5_0	4716523	11	5	0 dB

Table 37 Costas signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 37), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. FSK_PSK_C_1_15_11_5_s

This FSK/PSK Costas code signal was generated with the parameters described in Table 38. The frequency sequence is the sequence number 1 (4-7-1-6-5-2-3 kHz), 11 bits Barker code with 5 cycles per bit.The bandwidth used in each one of the carry frequencies is kept constant; therefore, the number of cycle per bit must vary as shown in Equations (3.32) and (3.34). As a consequence of keeping the same bandwidth for each frequency, the code period and the time spent by the signal in each frequency is constant as demonstrated in (3.33)and (3.35).

For example the bandwidth for the signal with carrier frequency 1 kHz is

$$BW_{1KHz} = \frac{\text{Carrier frequency}}{\text{Cycles per bit}} = \frac{f_C}{cpp} = \frac{1000 \text{ Hz}}{5} = 200 \text{ Hz} \tag{3.32}$$

and the Barker code period is

$$T_C = \frac{(\text{Cycles per bit})(\text{No of bits})}{\text{Carrier frequency}} = \frac{(5)(11)}{1000} = 55 \text{ ms.} \tag{3.33}$$

The bandwidth for the signal with carrier frequency 7 kHz is

$$BW_{7KHz} = \frac{\text{Carrier Frequency}}{\text{Cycles per bit}} = \frac{f_C}{cpp} = \frac{7000 \text{ Hz}}{35} = 200 \text{ Hz} \tag{3.34}$$

and the Barker code period is

$$T_C = \frac{(\text{Cycles per bit})(\text{No of bits})}{\text{Carrier frequency}} = \frac{(35)(11)}{7000} = 55 \text{ ms.} \tag{3.35}$$

It can be inferred that since the code period is 55 ms, the transmission time in each frequency of the sequence is five times the code period (due to the signal was generated with 5 periods Barker code for each frequency [15]), giving a transmission time in each frequency of 275 ms. Therefore, since the Costas sequence has 7 different frequencies, the Costas code period is 7 times the transmission time per frequency, resulting in a total of 1.925 s.

Now, and according with the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB tree to obtain the different

output layers. In this manner, Figure 72 was created which show the output matrix from layer 9 with a frequency resolution of 14.734 Hz. A zoom of the same layer output matrix is shown in Figure 73, there the values of the Costas frequency sequence are identified, as well as the Costas code period and bandwidth, giving the same values than the original signal.

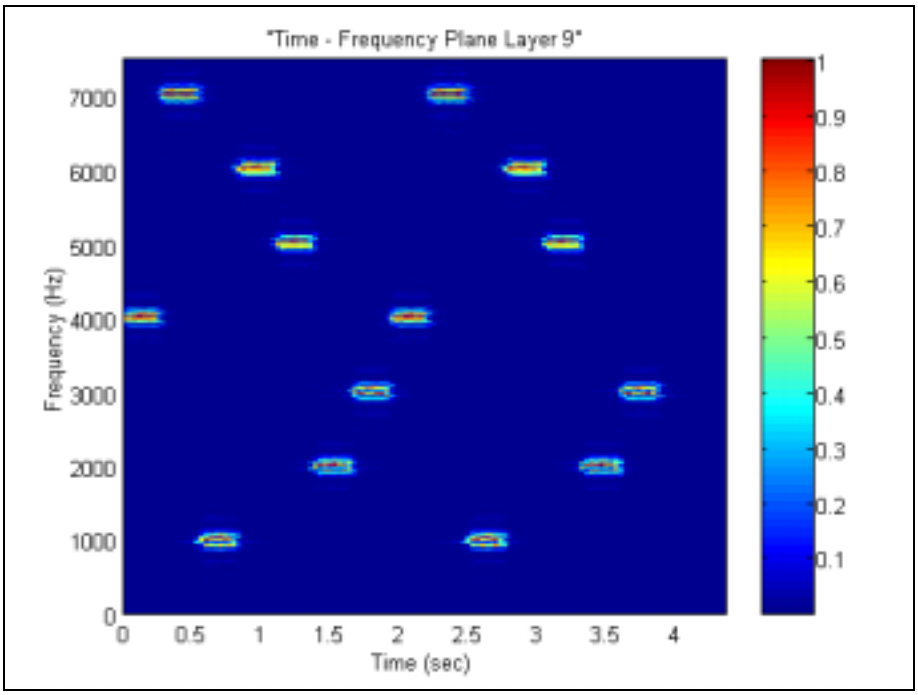

Figure 72 Output matrix from layer 9 of FSK_PSK_C_1_15_11_5_s (colorplot).

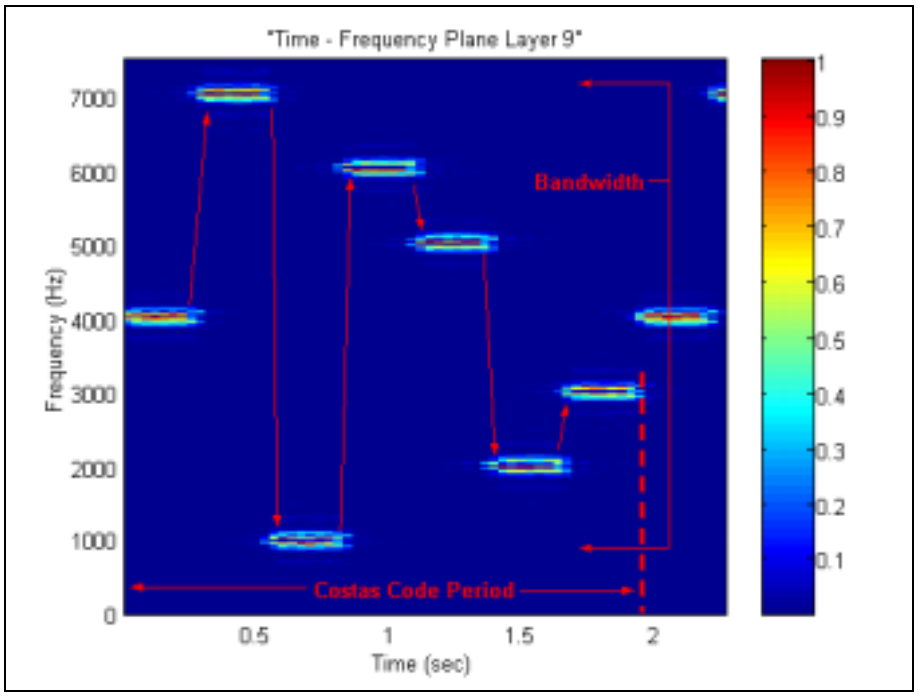

Figure 73 Zoom of output matrix from layer 9 of FSK_PSK_C_1_15_11_5_s (color-plot).

Examining a previous output matrix layer it is possible to achieve a better resolution in time and to see more details of the signals in each frequency of the sequence. Figure 74 shows the matrix output layer 8 in a contourplot marking the frequency sequence to be analyzed (4 kHz). Figure 75 shows a zoom of the same layer focusing in the 4 kHz frequency of the sequence for the second Costas code period. There the bandwidth (200 kHz), transmission time in the frequency (275 ms), and the Barker code period (55 ms) can be extracted. The 11 Barker bits are not visible in the plot because the resolution is not as high as desirable. Also, in the plot appears a modulation of 5 periods in the same carrier frequency indicating that the Barker code is repeated five times been this a setting in the Low Probability of Intercept Generator Program (LPIG [15]).

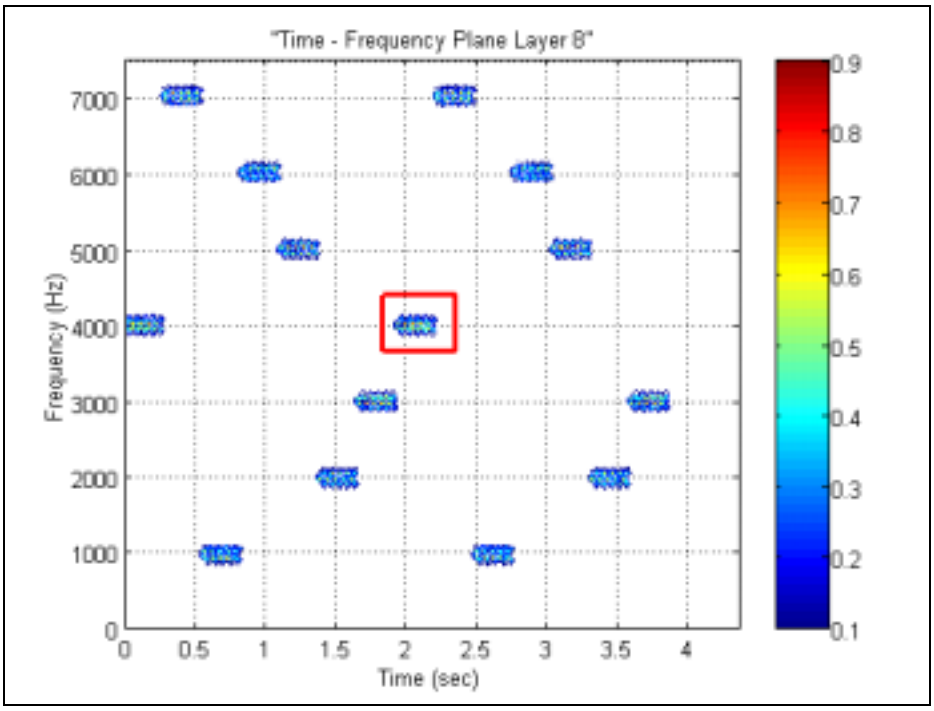

Figure 74 Output matrix from layer 8 of FSK_PSK_C_1_15_11_5_s (contourplot).

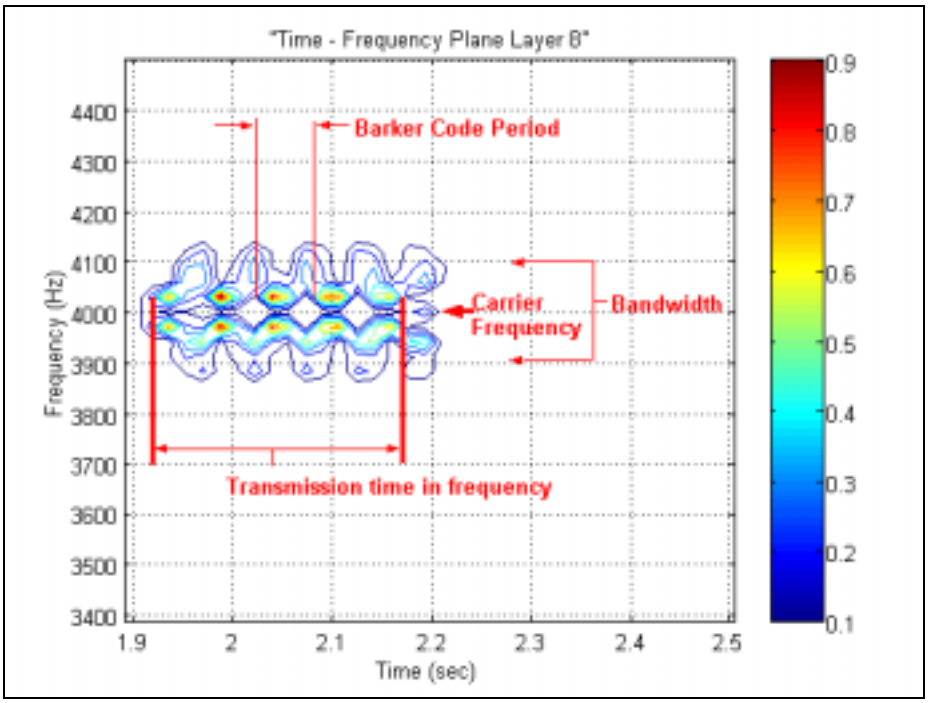

Figure 75 Zoom of output matrix from layer 8 of FSK_PSK_C_1_15_11_5_s (contourplot).

Table 38 presents a summary of the signal processing showing a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

FSK_PSK_C_1_15_11_5_s Parameters	**Generation**	**Detection**	**Comment**
Costas Sequence	4-7-1-6-5-2-3 kHz	4-7-1-6-5-2-3 kHz	
Sampling Frequency	15 kHz	15 kHz	Given
SNR	Only Signal	-	
Cycles per bit	5	-	
Barker Code Period	55 ms	55 ms	
Number of Bits per Barker Code	11	-	
Transmission time per Frequency	275 ms	275 ms	
Bandwidth per Carrier frequency (Barker)	200 Hz	200 Hz	
Costas Code Period	1.925 s	1.925 s	
Costas Bandwidth	6200 Hz	6306 Hz	106 Hz error

Table 38 Signal processing summary for FSK_PSK_C_1_15_11_5_s.

b. FSK_PSK_C_1_15_11_5_0

This FSK/PSK Costas code signal was generated with the parameters described in Table 39. It must be noted that this signal has almost the same parameters as the previous, the only difference is the SNR that now is 0 dB. Figure 76 was obtained, which provides a good time-frequency description of the evaluated signal from the output matrix at layer 9 when conducting the processing with the QMFB tree with a frequency resolution of 14.734. A zoom of the same layer output matrix is shown in Figure 77; the values of the Costas frequency sequence were identified, as well as the Costas code period and bandwidth, giving the same values than the original signal.

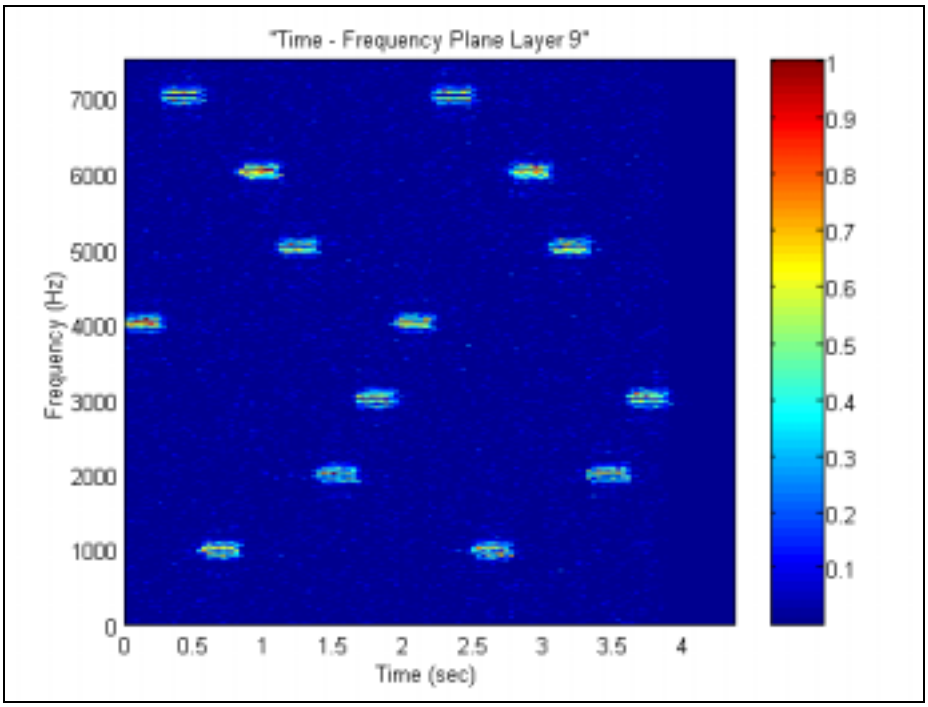

Figure 76 Output matrix from layer 9 of FSK_PSK_C_1_15_11_5_0 (colorplot).

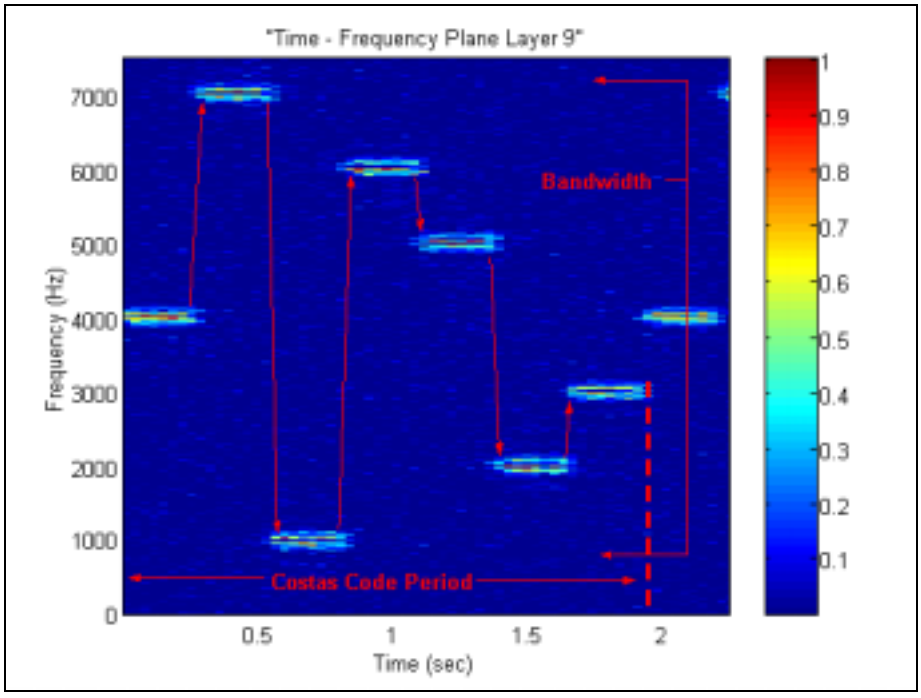

Figure 77 Zoom output matrix from layer 9 of FSK_PSK_C_1_15_11_5_0 (colorplot).

Examining a previous output matrix layer it is possible to achieve a better resolution in time and see more details of the signals in each frequency of the sequence. Figure 78 shows the matrix output layer 8 in a contourplot marking the frequency sequence to be analyzed (4 kHz). Figure 79 shows a zoom of the same layer focusing in the 4 kHz frequency of the sequence for the second Costas code period. The bandwidth (200 kHz), transmission time in the frequency (275 ms), and the Barker code period (55 ms) can be extracted. The 11 Barker bits are not visible in the plot because the resolution is not as high as desirable. Also, in the plot appears a modulation of 5 periods in the same carrier frequency indicating that the Barker code is repeated five times, been a setting in the LPIG [15].

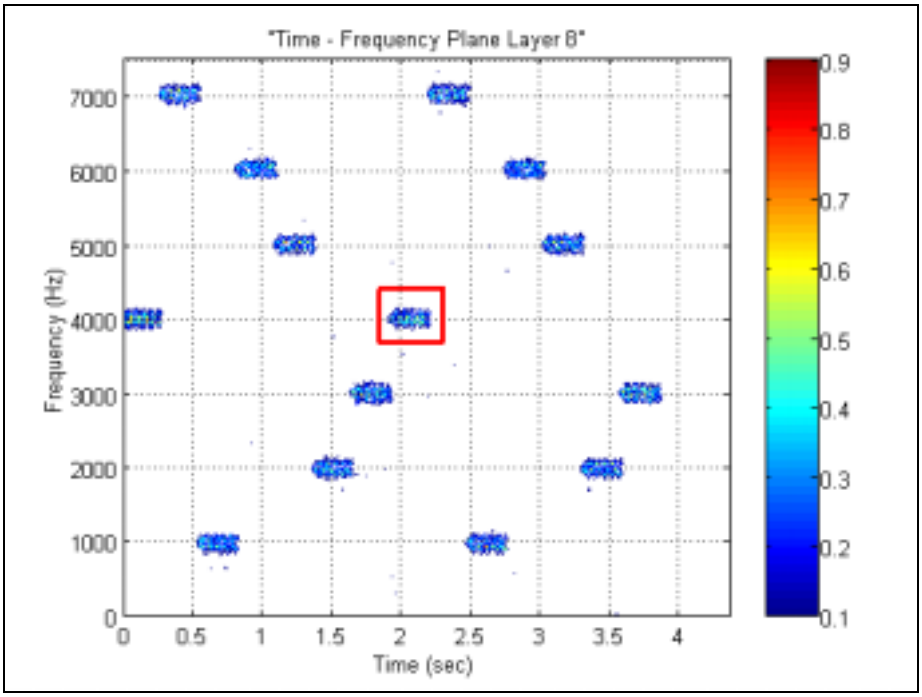

Figure 78 Output matrix from layer 8 of FSK_PSK_C_1_15_11_5_0 (contourplot).

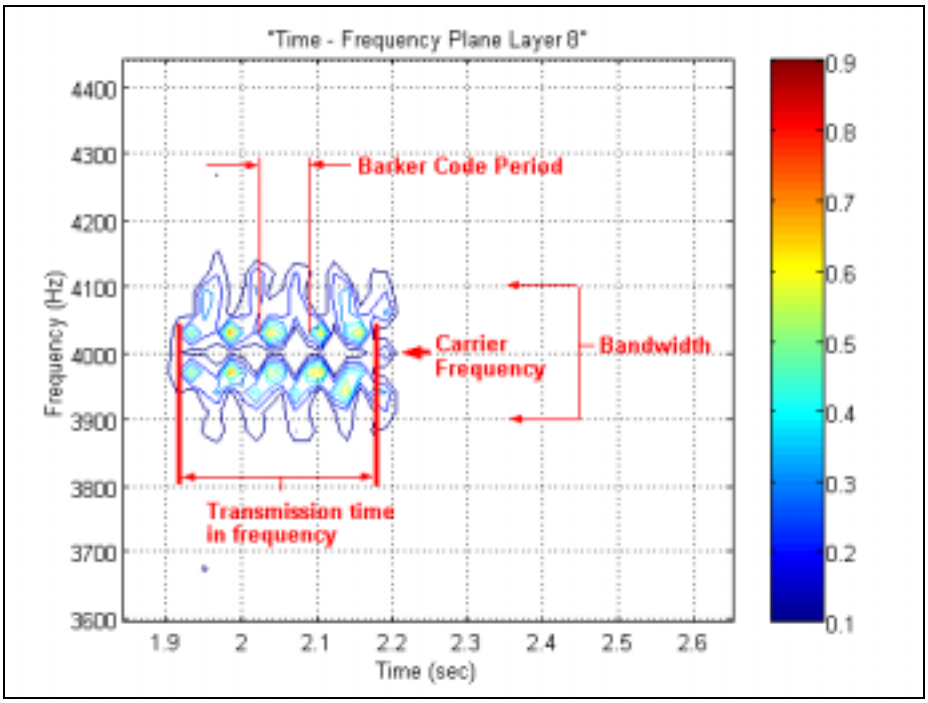

Figure 79 Zoom of output matrix from layer 8 of FSK_PSK_C_1_15_11_5_0 (contour-plot).

Table 39 presents a summary of the signal processing showing a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

FSK_PSK_C_1_15_11_5_0 Parameters	**Generation**	**Detection**	**Comment**
Costas Sequence	4-7-1-6-5-2-3 kHz	4-7-1-6-5-2-3 kHz	
Sampling Frequency	15 kHz	15 kHz	Given
SNR	0 dB	-	
Cycles per bit	5	-	
Barker Code Period	55 ms	55 ms	
Number of Bits per Barker Code	11	-	
Transmission time per Frequency	275 ms	275 ms	
Bandwidth per Carrier frequency (Barker)	200 Hz	200 Hz	
Costas Code Period	1.925 s	1.925 s	
Costas Bandwidth	6200 Hz	6306 Hz	106 Hz error

Table 39 Signal processing summary for FSK_PSK_C_1_15_11_5_0.

K. FSK/PSK COMBINED WITH TARGET-MATCHED FREQUENCY HOPPING

1. Brief Description

Instead of spreading the energy of the signal equally over a broad bandwidth, this type of technique concentrates the signal energy in specific spectral locations of most importance for the radar and its typical targets, within the broad-spectrum bandwidth. The produced signals have a pulse compression characteristic and therefore they can achieve low probability of intercept.

The implementation starts with a simulated target time radar response. This data is then Fourier transformed and the correspondent frequencies and initial phases are calculated using built-in Matlab functions in LPIG [17]. A random selection process chooses each frequency (between 64 different frequencies) with a probability distribution function defined by the spectral characteristics of the target of interest obtained from the FFT, so that frequencies at the spectral peaks of the target (highest magnitudes) are transmitted more often. Each frequency hop, transmitted during a specific period of time, is also modulated in phase, having its initial phase value modified by a pseudo-random spreading phase sequence code of values equally likely to be zero or π radians.

The matched FSK/PSK radar will then use a correlation receiver with a phase mismatched reference signal instead of a perfectly phase matched reference. This allows the radar to generate signals that can match a target's spectral response in both magnitude and phase.

Figure 80 illustrates the block diagram for the generation of FSK/PSK. This diagram and a deep description of the signal can be found in [17] and [18]. In addition Figures 81, 82, and 83 show the FSK/PSK target simulated response, the frequency components and frequency components histogram with number of occurrences per frequency for 256 frequency hops, respectively.

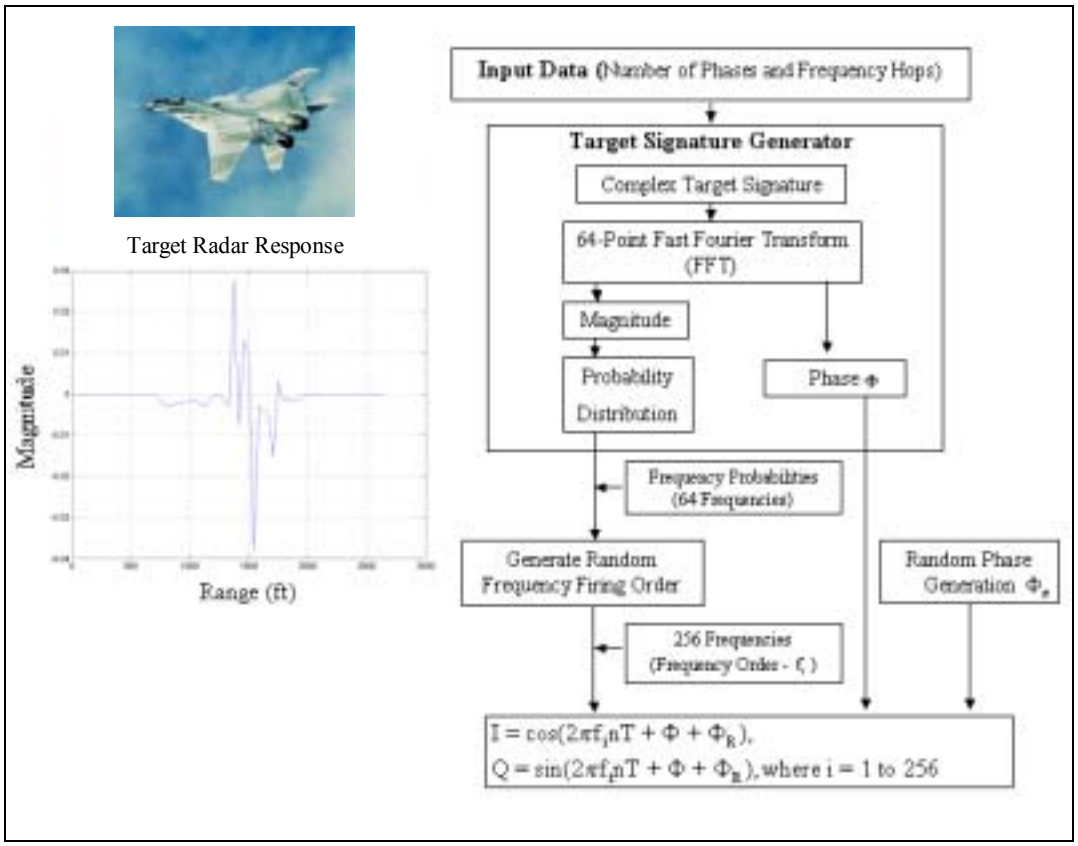

Figure 80 Block diagram of the implementation of the FSK/PSK Target matched waveform starting from the target radar response.

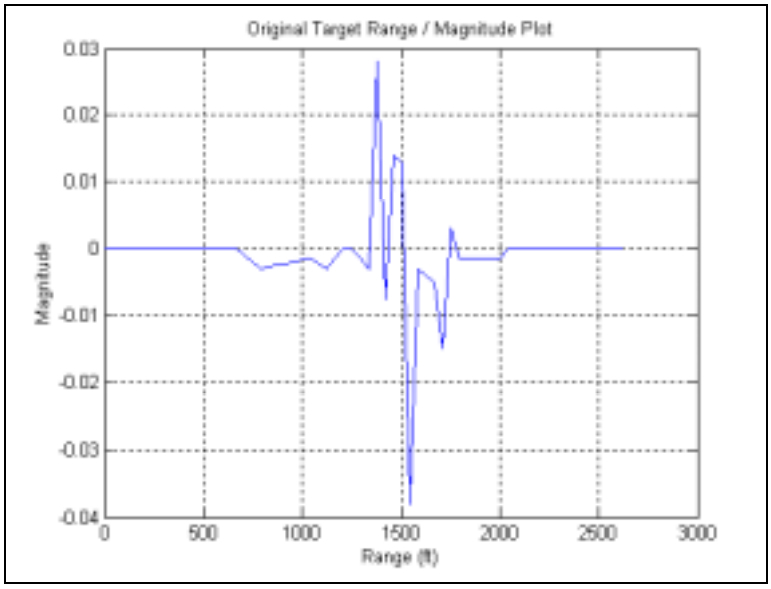

Figure 81 FSK/PSK Target simulated response.

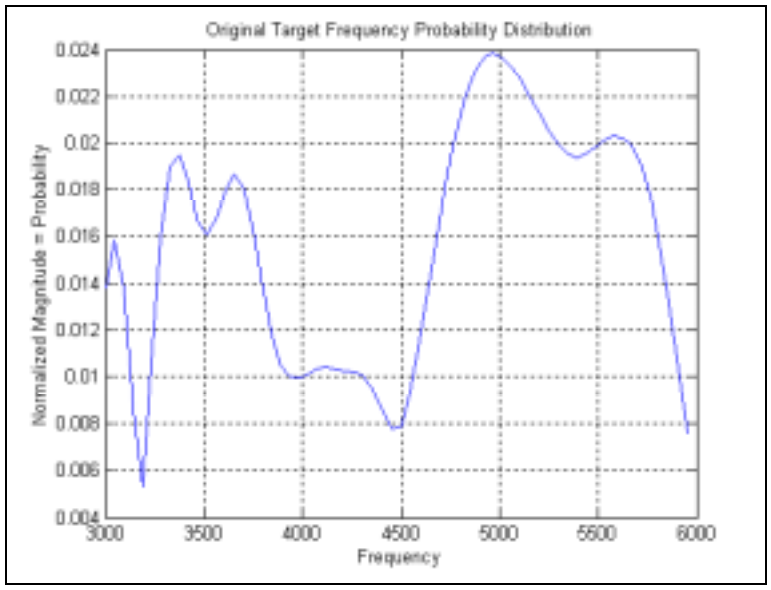

Figure 82 FSK/PSK Target frequency components and frequency probability distribution.

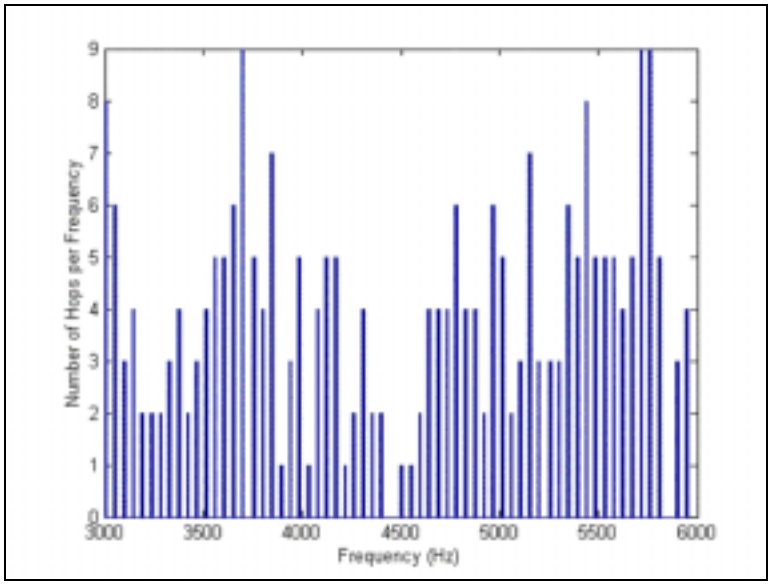

Figure 83 FSK/PSK Target frequency components histogram with number of occurrences per frequency for 256 frequency hops.

2. Processing FSK/PSK Target Code Signals with QMFB Tree

The FSK/PSK Target code signals to be worked with are given in Table 40. All the signals were generated with 5 or 10 different phases and 128 or 256 random frequencies (from the original 64 frequencies), the sampling frequency was 15000 Hz, the num-

ber of cycles per phase and the number of phases have the same value in the signal (since the cycle per phase is 1, the total amount of cycles is directly related with the number of phases), and the SNR are variable parameters.

	FSK/PSK Target	Number of random hops in frequency	Number of cycles and phases	SNR
1	FSK_PSK_T_15_128_5_s	128	5	Signal Only
2	FSK_PSK_T_15_128_5_0	128	5	0 dB
3	FSK_PSK_T_15_128_10_s	128	10	Signal Only
4	FSK_PSK_T_15_128_10_0	128	10	0 dB
5	FSK_PSK_T_15_256_5_s	256	5	Signal Only
6	FSK_PSK_T_15_256_5_0	256	5	0 dB
7	FSK_PSK_T_15_256_10_s	256	10	Signal Only
8	FSK_PSK_T_15_256_10_0	256	10	0 dB

Table 40 FSK/PSK Target signals to be processed by QMFB Tree.

From the list of signals already processed by the QMFB (in Table 40), only one of them is shown next. The rest of the signals can be seen in a technical report that will be published soon [21].

a. FSK_PSK_T_15_128_10_s

This FSK/PSK Target code signal was generated with the parameters described in Table 41, with 10 cycles, 10 phases, and 128 random frequencies. Now, and according with the description of the code given in Section C.1 in Chapter III, the signal is processed as an input to the QMFB Tree to obtain the different output layers. In this manner Figure 84 and Figure 85 were created which show the output matrix from layer 7 in color and contour plot, respectively. As a result of the particularly efficient LPI characteristics of this modulation, there is not much information that can be extracted from the plots in addition to the bandwidth and the concentration of the power in frequency. The random presentation of frequencies in the signal gives the impression of containing only noise. That way it can be observed a kind of noise in the signal and the power is mostly concentrated between 3 and 6 kHz as expected.

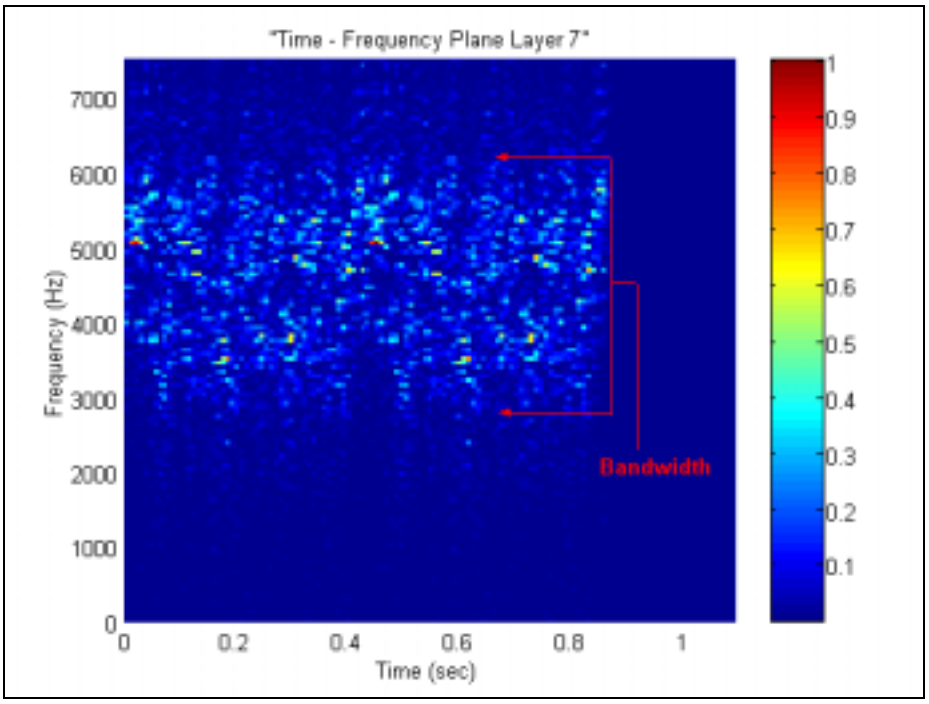

Figure 84 Output matrix from layer 7 of FSK_PSK_T_15_128_10_s (colorplot).

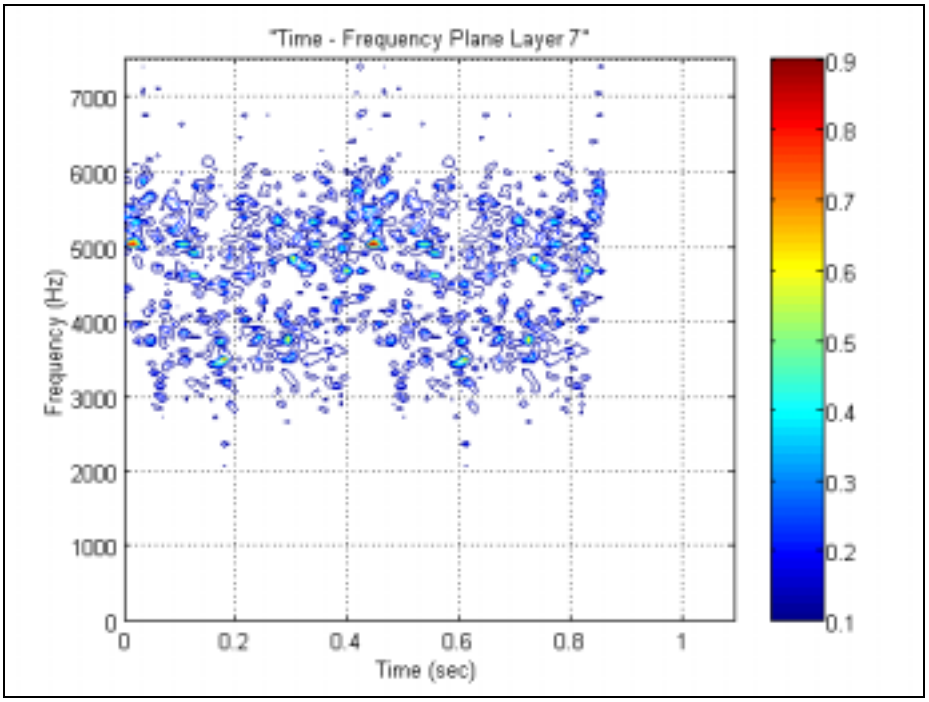

Figure 85 Output matrix from layer 7 of FSK_PSK_T_15_128_10_s (contourplot).

Table 41 presents a summary of the signal processing demonstrating a comparison of the real signal parameters versus the parameters extracted by the QMFB tree.

FSK_PSK_T_15_128_10_s Parameters	**Generation**	**Detection**	**Comment**
Sampling Frequency	15 kHz	15 kHz	Given
Number of Phases and cycles	10	-	
Number of Frequencies	64	-	
Number of Hops	128	-	
Bandwidth	3000 Hz	3000 Hz	
SNR	Only Signal	-	

Table 41 Signal processing summary for FSK_PSK_T_15_128_10_s.

b. FSK_PSK_T_15_128_10_0

This FSK/PSK Target code signal was generated with the parameters described in Table 42. This signal has almost the same parameters as the previous signal. The only difference is the SNR that now is 0 dB. As the signal is too noisy, the noise hides the signal and spreads the bandwidth making it extremely difficult to detect the FSK_PSK_T_15_128_10_0 signal parameters and extract some features. It is even harder to try to extract the code period.

In Table 42 a summary of the signal processing is shown, demonstrating a comparison of the real signal parameters versus the parameters detected by the QMFB tree.

FSK_PSK_T_15_128_10_0 Parameters	**Generation**	**Detection**	**Comment**
Sampling Frequency	15 kHz	-	Given
Number of Phases and cycles	10	-	
Number of Frequencies	64	-	
Number of Hops	128	-	
Bandwidth	3000 Hz	-	
SNR	0 dB	-	

Table 42 Signal processing summary for FSK_PSK_T_15_128_10_0.

L. ANALYSIS AND SUMMARY OF THE DIFFERENT SIGNALS PROCESSING RESULTS

After been processing all the LPI signals listed in Appendix B with the QMFB tree and analyzing the results it is possible to summarize the performance of the QMFB in detecting, displaying, and extracting features from the signals.

1. Binary Phase Shift Keying

As was expected, the QMFB tree revealed a very good performance in detecting the carrier frequency of the BPSK signals. The processing had no major difficulty in detecting the carrier frequency of signals with 0 or –6 dB of noise. With respect to the bandwith the QMFB tree shows that the extraction of this feature presents the same performance as the carrier frequency detection. The cycles per phase were not detected by the processing due to the resolution of the diagrams not being high enough to do this. Extracting the Code period the processing showed a good performance, but as for signals with high levels of noise (–6 dB) the resolution did not allow extracting this feature (no yellow bar for this feature in the bar plot below). Figure 86 shows a summary chart of the performance of the QMFB to detect carrier frequency, bandwidth, code period and bits in Barker code in the BPSK signals analyzed previously. This figure presents a comparison of the parameters in three different environments: signal only (blue), SNR= 0 dB (red) and SNR= –6 dB (yellow). The percentage in the chart describes an average of how close the extracted values are to the theoretical values.

	Carrier freq.(Hz)	Bandwidth (Hz)	Code period (ms)	Bits/code
Only signal	106%	106%	106%	50%
0 dB	101%	101%	82%	0%
(-) 6 dB	106%	106%	0%	0%

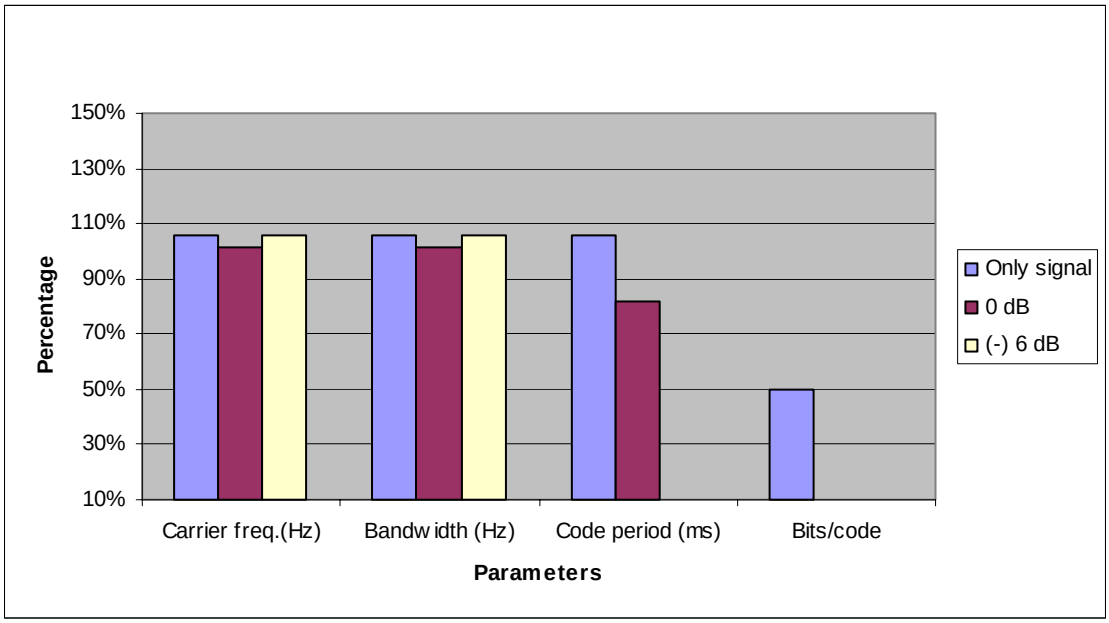

Figure 86 Performance of the QMFB processing detecting BPSK signals.

2. Frequency Modulation Continuous Wave (FMCW)

For this signal the QMFB tree show a very good resolution in detecting the carrier frequency under any SNR condition. The modulation bandwidth extraction gives an error incrementing the real value as the SNR goes to negative. The modulation period extraction shows a better detection for signals generated with longer modulation period (30 ms) than shorter ones (20 ms) and higher modulation bandwidth (500 Hz), but always has a problem extracting this feature from signals with SNR= –6 dB. Figure 87 shows a summary chart of the performance of the QMFB to detect carrier frequency, modulation bandwidth and modulation period in the FMCW signals analyzed previously. This figure presents a comparison of the parameters in three different environments: signal only (blue), SNR= 0 dB (red) and SNR= –6 dB (yellow). The percentage in the chart describes an average of how close the extracted values are from the theoretical values.

	Carrier freq.(Hz)	Mod.Bandwidth (Hz)	Mod.Period(ms)
Only signal	100%	114%	131%
0 dB	100%	131%	131%
(-) 6 dB	100%	153%	32%

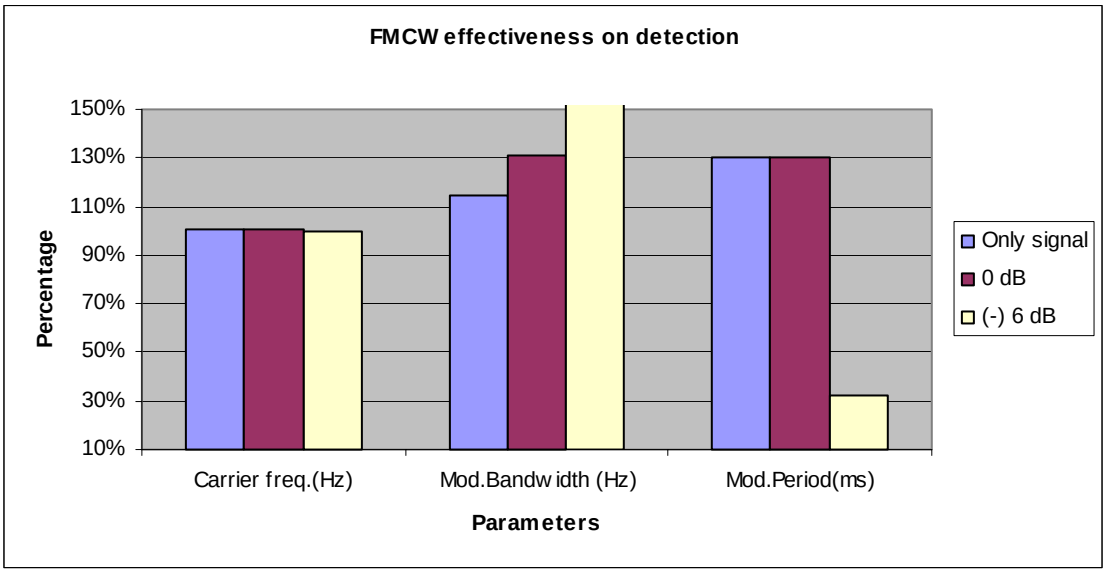

Figure 87 Performance of the QMFB processing detecting FMCW signals

3. Frank Code

The processing of this kind of signal with the QMFB tree gives good resolution extracting the carrier frequency, code period, and the number of phases for signals without noise, but the extraction of these parameters for noisy signals is difficult, specially obtaining the number of phases. Extracting the bandwidth of the signal was more accurate giving good results in most of the processing, except for the –6 dB condition. Figure 88 shows a summary chart of the performance of the QMFB to detect carrier frequency, bandwidth, code period and number of phases in Frank-coded signals analyzed previously. This figure presents a comparison of the parameters in three different environments: signal only (blue), SNR= 0 dB (red) and SNR= –6 dB (yellow). The percentage in the chart describes an average of how close the extracted values are from the theoretical values.

	Carrier freq.(Hz)	Bandwidth (Hz)	Code period (ms)	Phases
Only signal	102%	116%	104%	100%
0 dB	102%	107%	104%	0%
(-) 6 dB	75%	77%	78%	0%

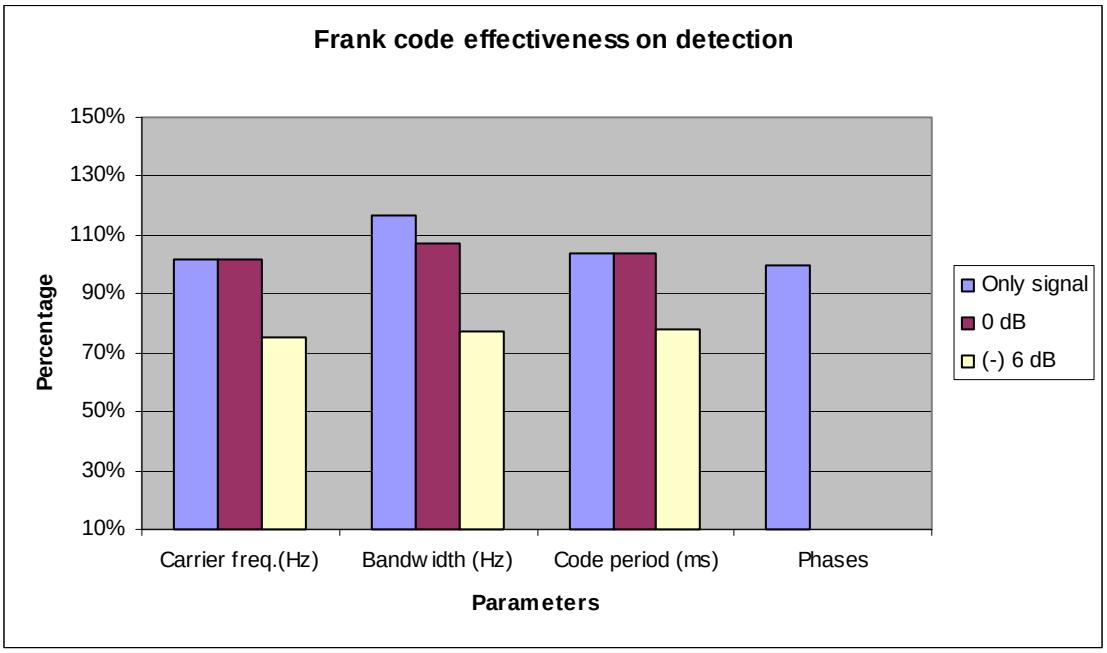

Figure 88 Performance of the QMFB processing detecting Frank-coded signals.

4. P1 Polyphase Code

Processing this signals, the QMFB detected the carrier frequency, the bandwidth, and the code period with very good resolution for the case of signal only and SNR=0 dB, but under SNR= –6 dB, the efficiency of the QMFB goes down in a 50% approximately. The number of phases can be extracted just in signals without noise. Figure 89 shows a summary chart of the performance of the QMFB to detect carrier frequency, bandwidth, code period and number of phases in P1-coded signals analyzed previously. This figure presents a comparison of the parameters in three different environments: signal only (blue), SNR= 0 dB (red) and SNR= –6 dB (yellow). The percentage in the chart describes an average of how close the extracted values are from the theoretical values. After processing the signals, the plots provide the values for carrier frequency, bandwidth and code period in any environment analyzed.

	Carrier freq.(Hz)	Bandwidth (Hz)	Code period (ms)	Phases
Only signal	102%	116%	104%	50%
0 dB	102%	113%	104%	0%
(-) 6 dB	50%	56%	52%	0%

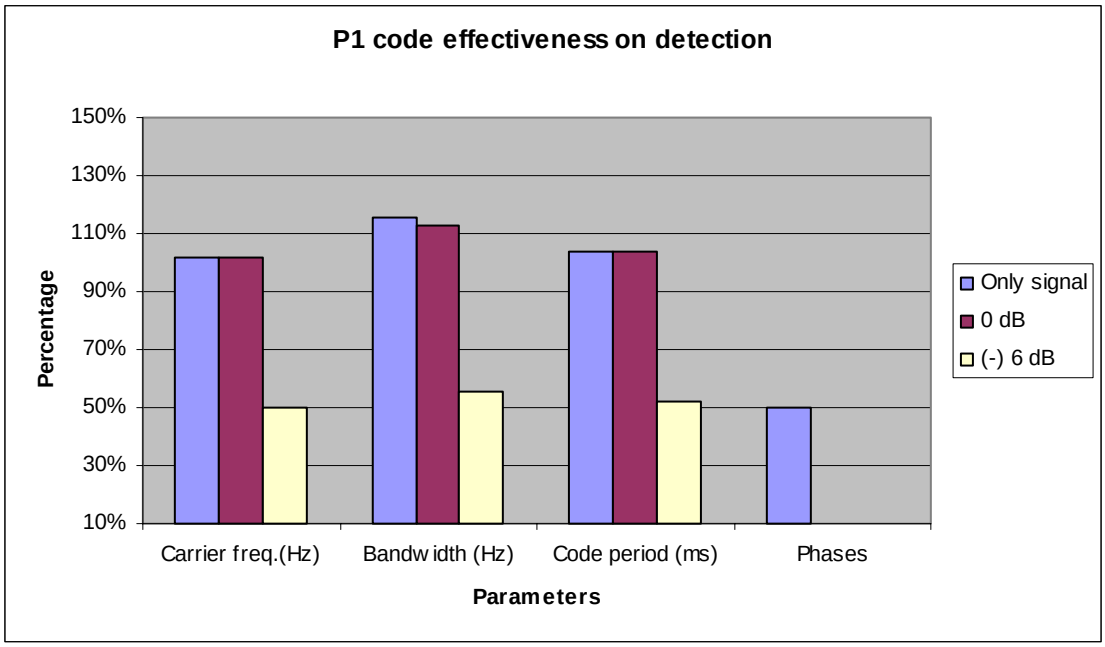

Figure 89 Performance of the QMFB processing detecting P1-coded signals.

5. P2 Polyphase Code

Processing this signals, the QMFB detect the carrier frequency, the bandwidth, and the code period with very good resolution for the case of signal only and SNR= 0 dB, but under SNR= –6 dB, the efficiency of the QMFB goes down in a 50% approximately. The number of phases can be extracted just in signals without noise. Figure 90 shows a summary chart of the performance of the QMFB to detect carrier frequency, bandwidth, code period and number of phases in P2-coded signals analyzed previously. This figure presents a comparison of the parameters in three different environments: signal only (blue), SNR= 0 dB (red) and SNR= –6 dB (yellow). The percentage in the chart describes an average of how close the extracted values are from the theoretical values. After processing the signals, the plots provide very accurate values for carrier frequency, and bandwidth, but not code period.

	Carrier freq.(Hz)	Bandwidth (Hz)	Code period (ms)	Phases
Only signal	102%	109%	98%	50%
0 dB	102%	100%	98%	0%
(-) 6 dB	50%	49%	47%	0%

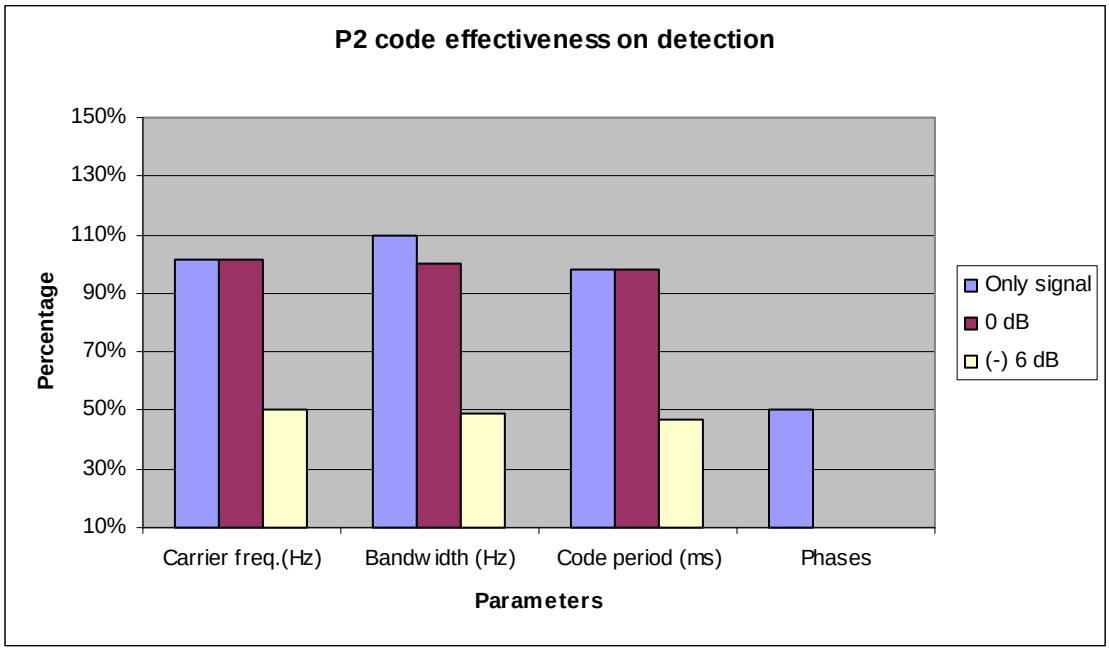

Figure 90 Performance of the QMFB processing detecting P2-coded signals

6. P3 Polyphase Code

Processing this signals with the QMFB gave almost the same results as the P2 and P1 signals showing an excellent job detecting the carrier frequency, bandwidth, and code period for signals without noise or SNR= 0 dB, but a very bad under SNR= –6 dB. The number of phases can be extracted just in signals without noise. Figure 91 shows a summary chart of the performance of the QMFB to detect carrier frequency, bandwidth, code period and number of phases in P3-coded signals analyzed previously. This figure presents a comparison of the parameters in three different environments: signal only (blue), SNR= 0 dB (red) and SNR= –6 dB (yellow). The percentage in the chart describes an average of how close the extracted values are from the theoretical values. After processing the signals, the plots provide very accurate values for carrier frequency, bandwidth and code period in any environment analyzed with a lower performance when SNR is less than –6 dB. The number of phases can be only extracted when noise is not presented and a number of cycles per phase is greater than 5.

	Carrier freq.(Hz)	Bandwidth (Hz)	Code period (ms)	Phases
Only signal	102%	112%	104%	75%
0 dB	99%	113%	106%	0%
(-) 6 dB	75%	78%	83%	0%

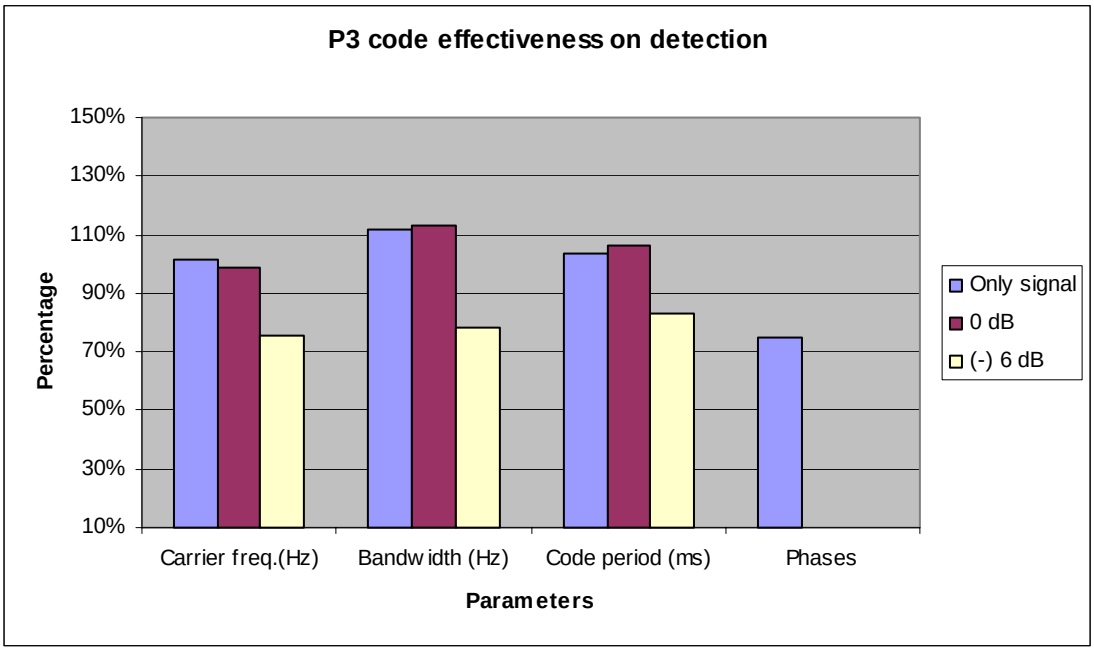

Figure 91 Performance of the QMFB processing detecting P3-coded signals.

7. P4 Polyphase Code

This signals demonstrated that the QMFB gave almost the same results as the other polyphase code signals showing an excellent job detecting the carrier frequency, bandwidth, and code period for signals without noise or SNR= 0 dB, but a very bad under SNR= –6 dB. The number of phases can be extracted just in signals without noise. Figure 92 shows a summary chart of the performance of the QMFB to detect carrier frequency, bandwidth, code period and number of phases in P4-coded signals analyzed previously. This figure presents a comparison of the parameters in three different environments: signal only (blue), SNR=0 dB (red) and SNR= –6 dB (yellow). The percentage in the chart describes an average of how close the extracted values are from the theoretical values. After processing the signals, the plots provide very accurate values for carrier frequency, bandwidth and code period in any environment analyzed with a lower performance when SNR is less than –6 dB. The number of phases can be only extracted when noise is not presented and a number of cycles per phase is greater than 5.

	Carrier freq.(Hz)	Bandwidth (Hz)	Code period (ms)	Phases
Only signal	102%	116%	104%	50%
0 dB	102%	113%	104%	0%
(-) 6 dB	75%	81%	80%	0%

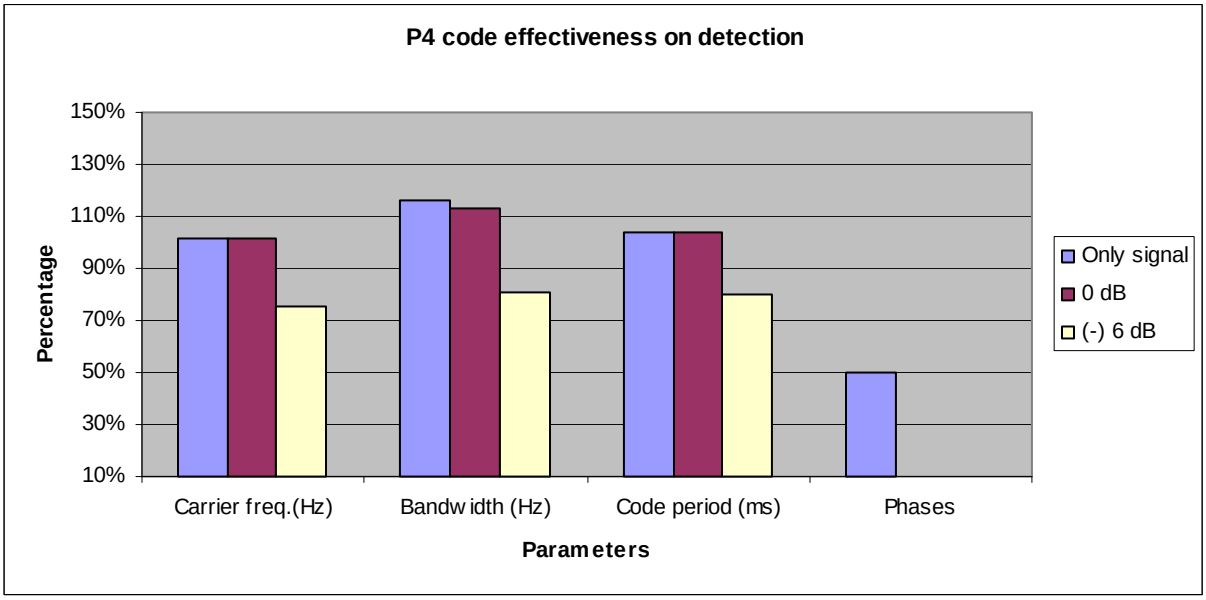

Figure 92 Performance of the QMFB processing detecting P4-coded signals.

8. Costas Code

This specific kind of signal when processed by the QMFB tree gave excellent frequency sequence detection with the best resolution measuring the transmission time per frequency. The code period was detected with a very good resolution for all the signals and the same happened for the bandwidth. Figure 93 shows a summary chart of the performance of the QMFB to detect carrier frequency, bandwidth, and code period in Costas-coded signals. This figure presents a comparison of the parameters in three different environments: signal only (blue), SNR= 0 dB (red) and SNR= –6 dB (yellow). The percentage in the chart describes an average of how close the extracted values are from the theoretical values. After processing the signals, the plots provide very accurate values for carrier frequency, bandwidth and code period in any environment analyzed.

	Sequence	time in frequency(ms)	Code period (ms)
Only signal	100%	100%	100%
0 dB	100%	100%	100%
(-) 6 dB	100%	100%	100%

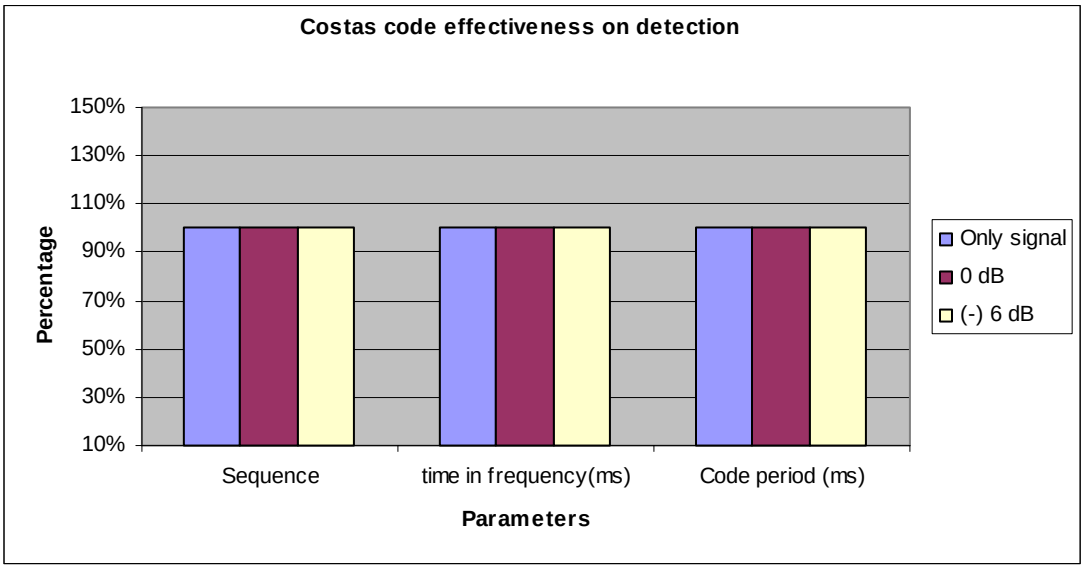

Figure 93 Performance of the signal processing detecting Costas-coded signals.

9. Frequency Shift Keying/Phase Shift Keying Combined with Costas Code (FSK/PSK Costas)

This very specific kind of signal has very important characteristics and applications as an LPI radar signal and the QMFB tree had a very good performance analyzing it. The frequency sequence was detected in every case with excellent resolution. The same situation happened in the detection of the Barker code period and the transmission time per frequency. The bandwidth per carrier frequency was detected with a good resolution, harder to see than in other signals, but clear enough to be defined. Figure 94 shows a summary chart of the performance of the QMFB to detect carrier frequency, bandwidth and code period in FSK/PSK Costas-coded signals analyzed previously. This figure presents a comparison of the parameters in two different environments: signal only (blue) and SNR= 0 dB (red), hence there are no yellow bars. The percentage in the chart describes an average of how close the extracted values are from the theoretical values. After processing the signals, the plots provide very precise values for carrier frequency, bandwidth and code period in any environment analyzed except for SNR lower than 0 dB where none of the parameters can be extracted.

	Sequence	Bandwidth (Hz)	Code period (ms)
Only signal	100%	99%	100%
0 dB	100%	98%	100%

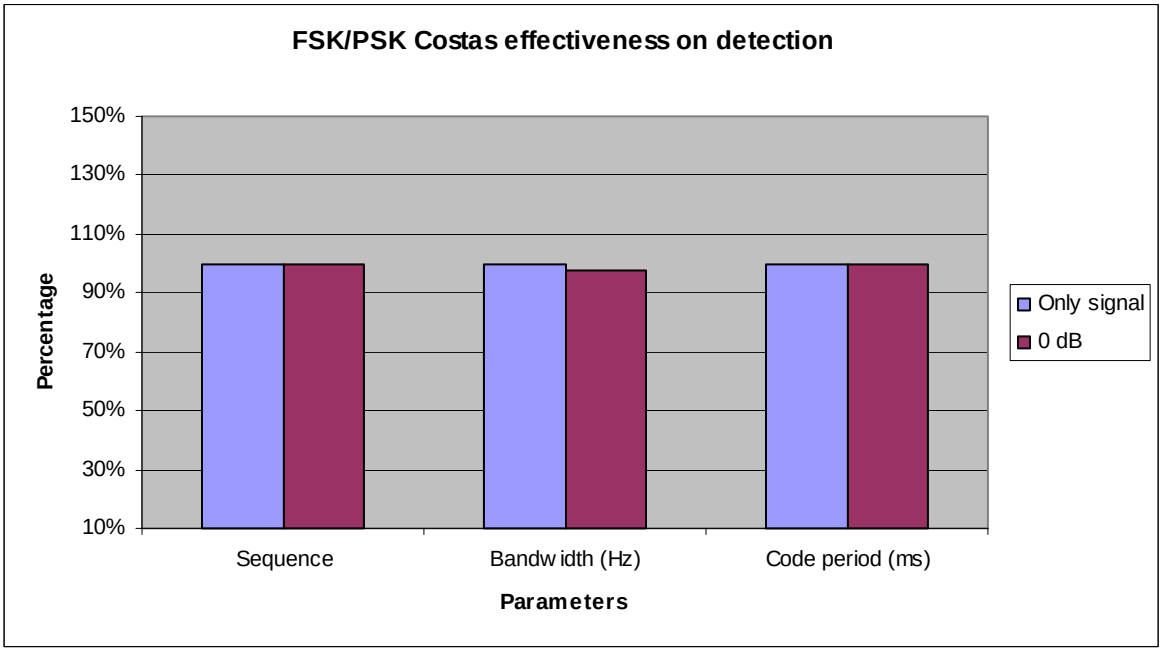

Figure 94 Performance of the QMFB processing detecting Costas-coded signals.

10. Frequency Shift Keying/Phase Shift Keying Combined with Target-Matched Frequency Hopping (FSK/PSK Target)

Processing this signals with the QMFB tree revealed that it was very hard to extract features other than the bandwidth. But since these signals were generated with 128 or 256 random jumps in frequency between 64 original and different ones from a target radar response, it is extremely hard to determine the frequency sequence. Therefore, the plots generated by the processing looks like a signal with a strong concentration of noise in a specific bandwidth, making it really hard to identify it as a FSK/PSK Target signal. Figure 95 shows a summary chart of the performance of the QMFB to detect carrier frequency, bandwidth and code period in FSK/PSK Target signals analyzed previously. This figure presents a comparison of the parameters in three different environments: signal only (blue), SNR= 0 dB (red) and SNR= –6 dB (yellow). The percentage in the chart describes an average of how close the extracted values are from the theoretical values. After processing the signals, the plots provide very precise values for bandwidth for signal-only

condition. The sequence and code period in any environment analyzed cannot be extracted.

	Sequence	Bandwidth (Hz)	Code period (ms)
Only signal	0%	96%	0%
0 dB	0%	0%	0%

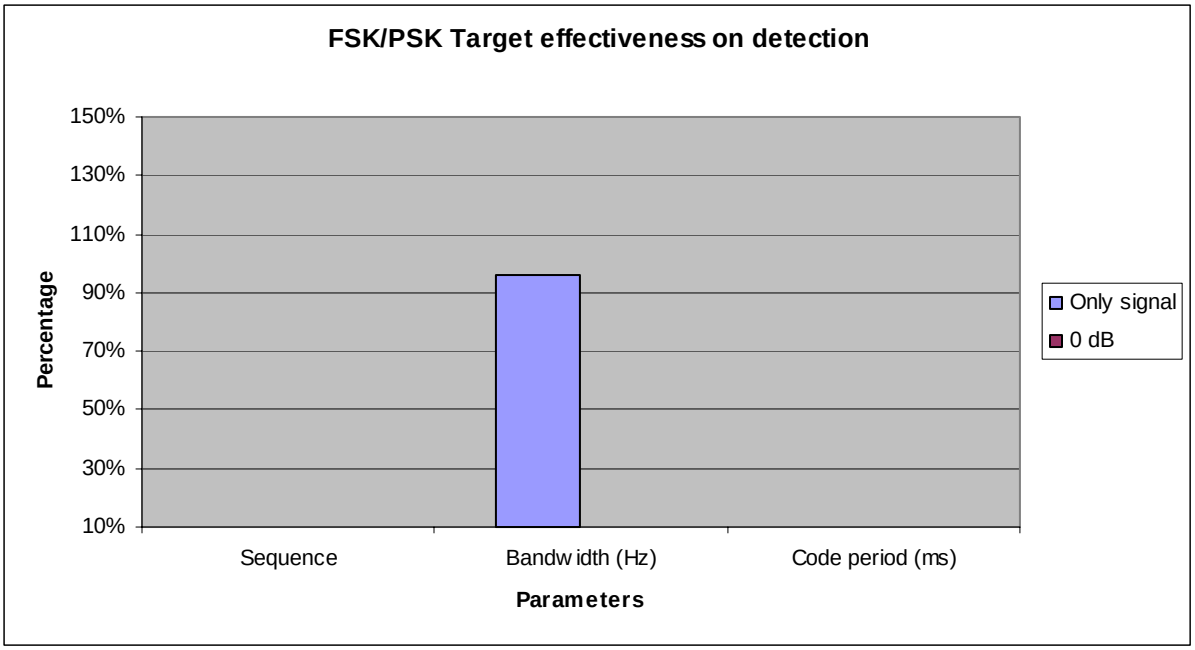

Figure 95 Performance of the QMFB processing detecting FSK/PSK target signals.

M. COMPARISON OF DIFFERENT POLYPHASE-CODED SIGNALS

The objective of comparing different polyphase-coded signals is to demonstrate the effectiveness of the proposed signal processing to discriminate among different polyphase modulations. One signal per each modulation was selected, having the same carrier frequency (1 kHz)), sampling frequency (7 kHz), number of phases (16) and cycles per phase (5). The most important deviation among the signal is related to the phase shift. The increment between phases produces diverse performance in the modulations, facilitating an accurate identification of each polyphase modulation.

Table 43 shows each one of the analyzed signals and their most important characteristics. In addition, this table presents the resulting phases for the generation of 16 phases (N= 4 or N^2= 16). The resulting figures after QMFB are provided along with the phase shift plots. The analysis of these plots reveals a close relationship where the conducts of the phases approximate to the frequency-time description of the signal.

Signal	**Phase shift values**	**Comment**
Frank	0 0 0 0 0 1.5708 3.1416 4.7124 0 3.1416 6.2832 9.4248 0 4.7124 9.4248 14.1372	Phase changes between adjacent codes are the smallest.
P1	0 –3.1416 6.2832 28.2743 –2.3562 –3.9270 7.0686 30.6305 –4.7124 –4.7124 7.8540 32.9867 –7.0686 –5.4978 8.6394	Lowest code increments from one code element to code element in the center of the waveform
P2	3.5343 1.1781 –1.1781 –3.5343 1.1781 0.3927 –0.3927 –1.1781 –1.1781 –0.3927 0.3927 1.1781 –3.5343 –1.1781 1.1781 3.5343	Symmetric at center frequency
P3	0 0.0123 0.0491 0.1104 0.1963 0.3068 0.4418 0.6013 0.7854 0.9940 1.2272 1.4849 1.7671 2.0739 2.4053 2.7612 (one row)	Largest code element to code element are on middle of the P3 code
P4	0 –2.9452 –5.4978 –7.6576 – 9.4248 –10.7992 –11.7810 –12.3700 –12.5664 –12.3700 –11.7810 –10.7992 –9.4248 –7.6576 –5.4978 –2.9452 (one row)	Largest code element to code element are on the two ends of the P4 code

Table 43 Different Polyphase-coded signals and differences for N^2=16.

1. Frank Code

The actual Frank-coded signal consists of a carrier (1 kHz), the place of which is modulated according to the indicated baseband waveform sequence. For each frequency or section of the step chirp, a phase group consisting of N phase samples is obtained and the total number of code phases is N^2. Note that the phase increments within the four phase groups are 0˚, 90˚, 180˚, and 270˚. However, the phases of the last group are ambiguous (>180˚) and appear as –90˚ phase steps. The last phase group, because of the ambiguity, appears to complete one 360˚ (Figure 96).

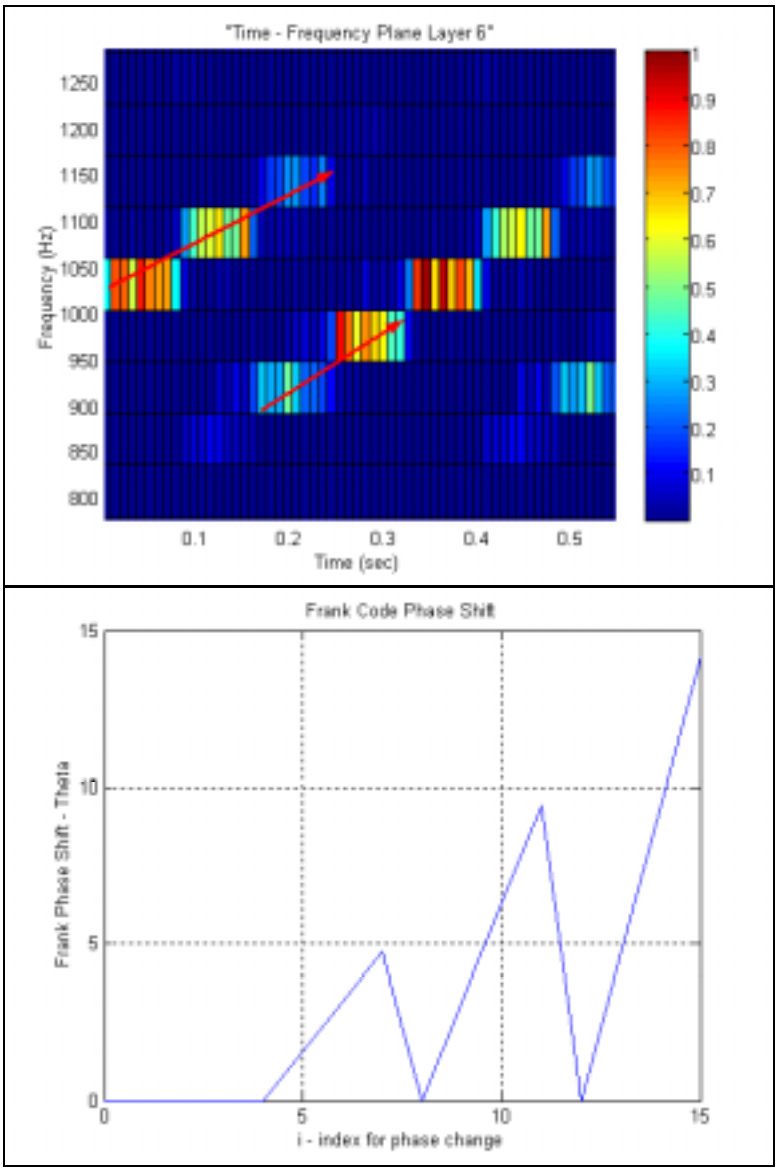

Figure 96 Frank-coded signal resulting plot after QMFB and phase shift.

2. P1

This code has the lowest code increments from one code element to code element in the center of the waveform as shown in Figure 97. For an odd number of *N*, the resulting phases are the same as the Frank code except the phase groups are rearranged. For *N* even, P1 code has the same phase increments, within each phase group, as the P2 except the starting phases are different.

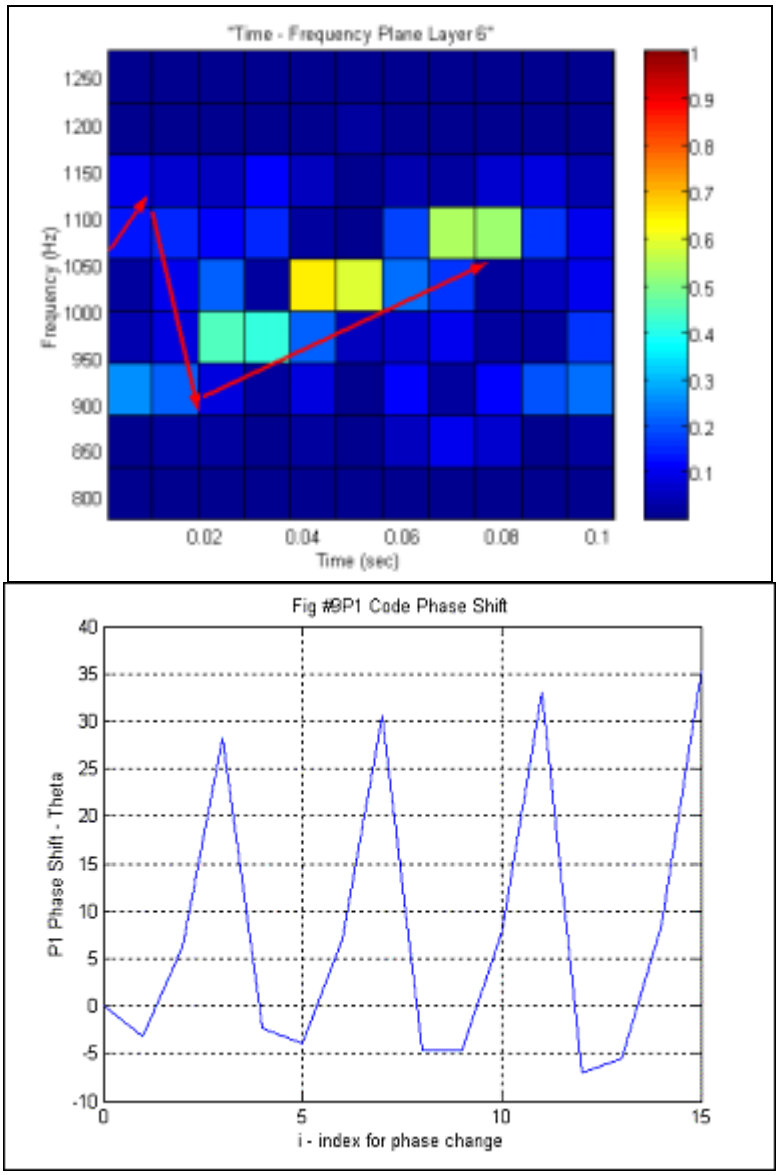

Figure 97 P1-coded signal resulting plot after QMFB and phase shift.

3. P2

This code is valid for even numbers of phases, and each group of the code is symmetric about zero phase as shown in Figure 98.

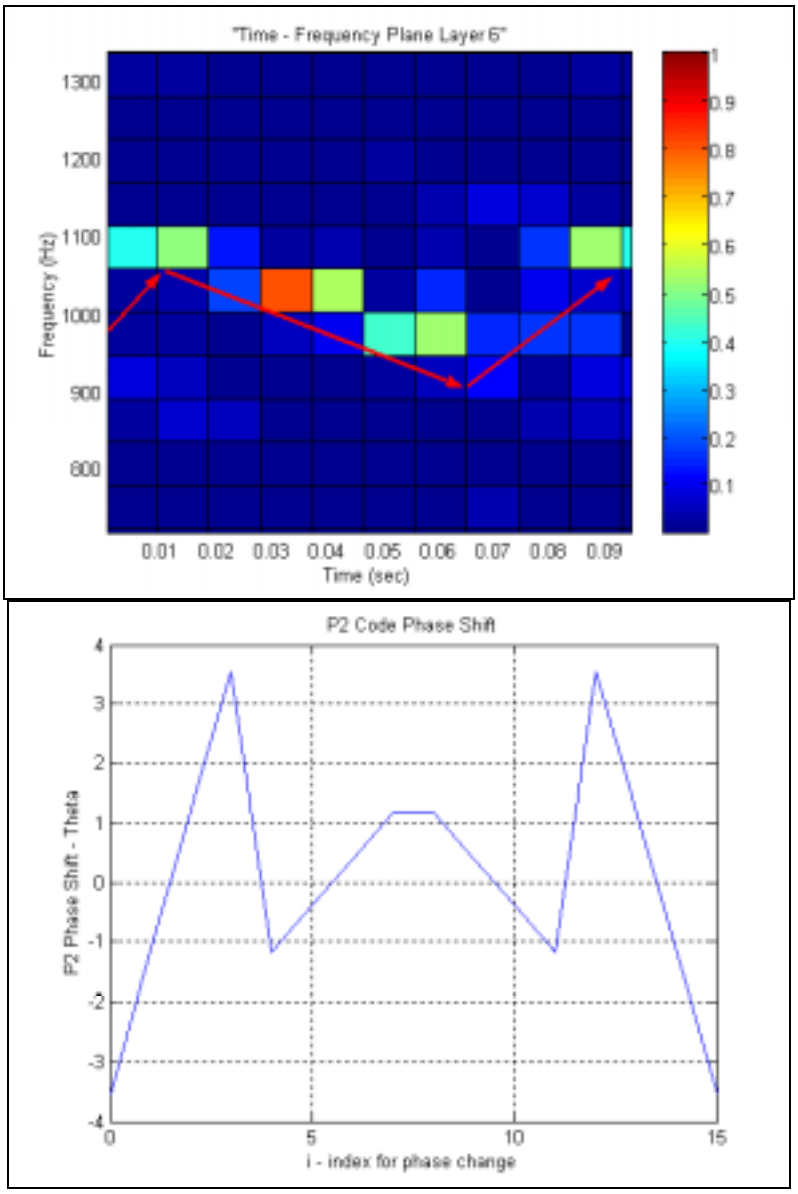

Figure 98 P2-coded signal resulting plot after QMFB and phase shift.

4. P3

P3 only differs from Frank code by 180° phase shifts every $N^{1/2}$ code elements (one frequency group) and by added phase increments that repeats every $N^{1/2}$ samples (every frequency group). The largest phase increments from code element to code element are in the middle of the P3 code as presented in Figure 99.

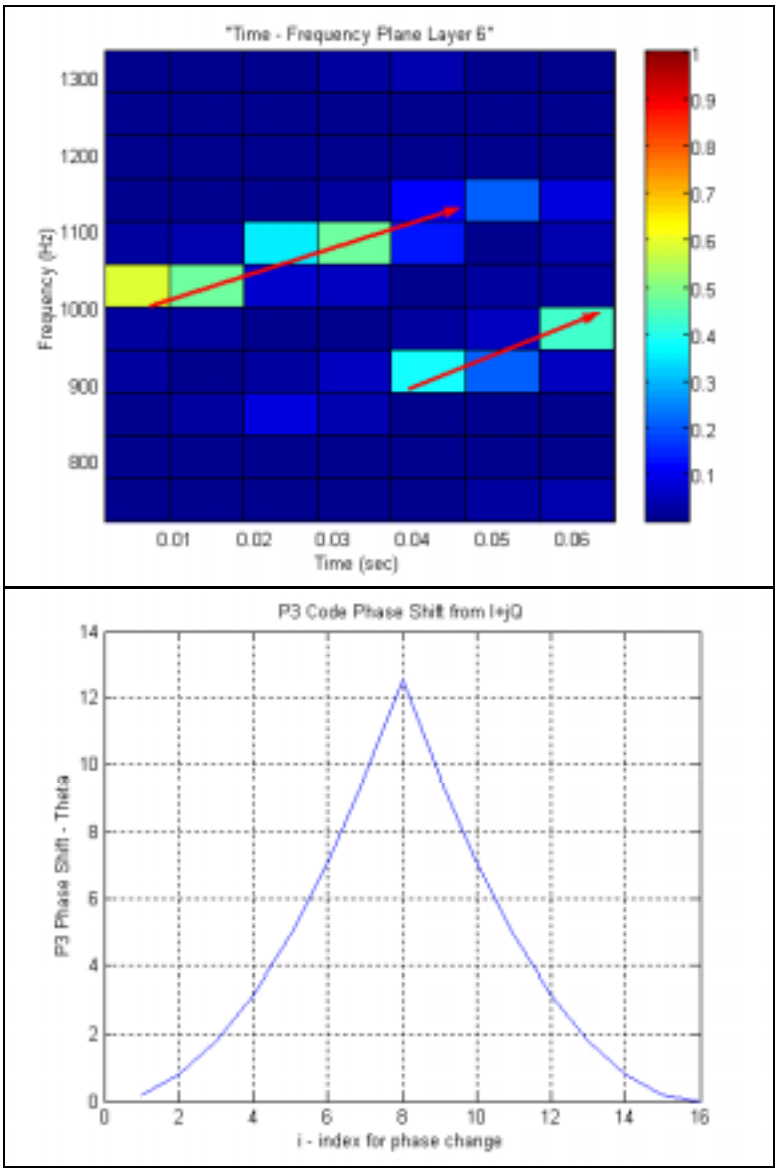

Figure 99 P3-coded signal resulting plot after QMFB and phase shift.

5. P4

P4 code is very similar to P1 code except that the phase samples are those of a sampled chirped waveform rather than step-chirp waveform. It is noted that the largest phase increment s from code element are on the two ends of the P4 code. Figure 100 describes the resulting signal and its phase shift.

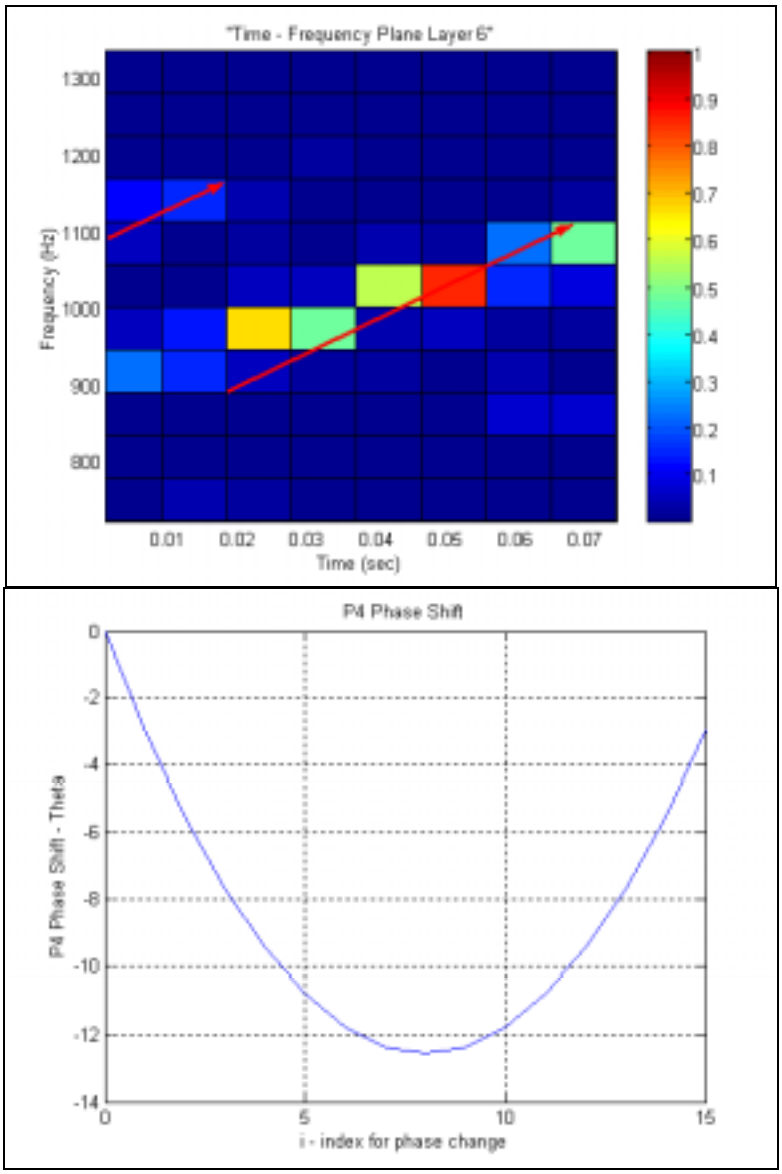

Figure 100 P4-coded signal resulting plot after QMFB and phase shift.

This chapter discussed the results of processing the LPI signals from the LPIG code as described in [15]. It finishes with an analysis and comparison of the different signals processing results, where the performances of the QMFB tree are specified for every class of signal depending on the results obtained. The next chapter gives conclusions and recommendations generated from this research.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The goal of this research is to use a method to decompose waveforms, using orthogonal basis functions and a quadrature mirror filter bank (QMFB) tree, and to extract detailed information about embedded LPI (spread spectrum) radar signals.

A list of different and well-known LPI radar signals were processed with the QMFB tree and the output layers were displayed and analyzed in a feature extraction procedure. Specifically, estimating the signal parameters such as bandwidth, center frequency, duration and position in time of the waveform were accomplished. The processing started by analyzing a tone signal and binary phase shift keying signals to show how the processing works, what kind of results were expected, and how to extract the parameters from the outputs of the QMFB tree.

Polyphases signals were processed obtaining very effective feature extraction results, where different parameters such as carrier frequency, bandwidth, and code period were extracted with good resolution for the case of signal only and SNR= 0 dB. In the case of SNR= –6 dB, the extraction of this parameters was difficult due to the noisy condition of the signals, but still gave results close enough to estimate the parameters. Related to the number of phases and the Barker code sequence, the detection of these parameters was possible for the case of signal only.

Costas Code signals were processed giving the best results as a feature extraction procedure. Parameters such as the sequence of the frequency hopping, the transmission time per frequency, and the code period were extracted with an effectiveness of 100% unaffected by the SNR condition of the signals.

FSK/PSK Costas Code signals were processed with the QMFB giving a very good effectiveness in detecting all the important parameters related to this signal (the sequence of the frequency hopping, the bandwidth, and the code period) in the case of signals without noise and SNR= 0 dB.

The FSK/PSK Target signal processing gave a poor result as a feature extraction procedure. The bandwidth was the only parameter that could be extracted with a good resolution, but just for the case of signal only. This is due mainly to the complexity of the generation of this kind of signal, where the plots generated by the QMFB processing looks like a signal with a strong concentration of noise in a specific bandwidth, making it really hard to identify it.

The overall architecture of the QMFB presented in this thesis, as shown in Figure 3, is a good alternative to an intercept receiver, particularly for detecting hopped LPI spread spectrum radar signals, as the FSK/PSK Costas processed in Chapter III.

The feature extraction gives a time-frequency plane with good estimation of position, center frequency and time duration of the signals processed, although the estimates of bandwidth degrade in poor SNR. With respect to the detection of phases and cycles per phase (or per bit, depending on the signal), there is much to do to improve the resolution, particularly in signals with some levels of noise.

This technique alone is not sufficient to process the multiplicity of available LPI waveforms, but the combined use of this technique with others, such as Wigner distribution [18], Cyclostationary processing [16], and Higher Order Statistics [15] will provide the desired capability.

The QMFB tree is demonstrated as a good tool to process any basic LPI radar signal and it can be used on any other type of signals to do the feature extraction and that way obtain a better understanding of the signal characteristics.

B. RECOMMENDATIONS

The QMFB tree must be improved to obtain a better resolution in detecting the phases and cycles per phase of the intercepted signals.

An algorithm to determine relevant signal cell energy features can analyze the output matrices from any layer of the QMFB. A list of these features could be sent to a classifier, which determines which cells belong to common transmitters, in order to output a list of transmitter types and parameters. Further, filtering could be done at this point

and thus eliminate probable false alarms and signals that are of no interest to the interceptor. Including a signal classifier in the analysis is an obvious next step. Especially in some particular scenarios, it would be interesting to see how well an adaptive classifier, such as an artificial neural network, could use the information output from an analyzer algorithm to detect and classify signals.

It is also instructive to compare the capabilities and performance of other methods to the one discussed here. It may be possible to combine some of the concepts presented here with some of these methods. For example, to look for a solution to the problem to extracting specific parameters in signals with added noise, a High Order Statistic (HOS) process at the output of every layer in the QMFB tree structure could be added and that way try to filter the noisy signals getting a better performance and resolution.

MATLAB® has been shown to be a very useful tool to develop the codes to work with the QMFB tree structure, but it takes some time to process the signals, particularly when the signals loaded are long. In consequence, some efforts must be focused on developing a more efficient code that takes less time to do the processing.

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix A: Matlab Programs and Functions

```
%*******************************************************************************
% startpoint.m
%               Main Control Program
%
% Use:          This is an m file to load the signal that will be filtered by the QMFBT.
%               The signal file name need to be written after load command and the
%               sampling frequency needs to be specified. The program considers the
%               signals storage in a folder called "LPI_Signal_Generator_II".
%
% Author:       Pedro Jarpa. Captain Chilean Air Force.
% Date:         July 30, 2002.
%
%*******************************************************************************
clear all;clc;
%_______________________________________________________________________________
% --Setting the signal to analize--
%   Write the signal directory and file name after load command in the next line
load C:\Jarpa\Thesis\LPI_Signal_Generator_II\FR_1_7_4_5_s;              % This is to load the signal from another directory
fs=input('What is the sampling frequency of the loadded signal (in Hz)?'); % This parametrer must be setted from the file loaded above
[v,u]=size(I);  % Since the QMFBT work with incoming signals as columns, this step is to transform
            % the incoming signal to a column vector if its a row vector
if v==1
   I=I';    % This apply if the incoming signal is a row vector
   Q=Q'; % This apply if the incoming signal is a row vector
end
%_______________________________________________________________________________
% -- Formating the Signal -- ( Note: This section apply only if the next section is comented/not used)
% This section must be used only in the case of analizing the signal received without padding zeros to make it a power of two.

% NUM_SAMPLES = 2 .^(log(length(I))/log(2));  % To generate a Test Signal of same number of samples as input Signal
% T=1/fs;
% signal=I+j*Q;
% tt=signal;
%_______________________________________________________________________________
% -- Formatting the Signal --
%
T=1/fs;                 % Period of the sampled signal
signal=I+j*Q;           % Forming the signal
L=length(signal);       % Length of the signal
Lc=ceil(log2(L));       % It will give the number of layers.
z=((2.^Lc) - L);        % In the case that the signal length does not correspond to a power of two
                % the signal will be padded with zeros until the next power of 2 (2^(number of layers))
zz=zeros(1,z);          % Vector of zeros to be padded to incoming signal to do it a power of two in length
tt=cat(1, signal, zz'); % New signal vector to be applyed to the QMFB resulting from concatenate the original signal with the zeros
NUM_SAMPLES = length(tt);% Length of the new signal
%_______________________________________________________________________________
% -- Filtering the Signal with the QMFB -- (qmfb function)
```

```
% qmfb(tt,'tsinc');      % Here the function qmfb is used, it will apply the filter bank to the signal with the
sinc modified filter
lay=qmfb(tt,'tsinc');    % Here the function qmfb is used, it will apply the filter bank to the signal with the
sinc modified filter
                % "lay" is set equal to the qmfb function to get the number of the last layer processed by
qmfb function
                % and that way use it to run the multiple set of plot.
% clear tt;          % Optional. Just to clear the signal.
%_____________________________________________________________________________________
% -- Plots --
%
% Loading the Layers
for yy=4:7
   load(['c:\Jarpa\Thesis\data\layer',num2str(yy)]) ;  % Setting the file directory and layer's number in the
load command
   % Formating the Layer
   layer = abs(R + j*Q);                    % Taking the absolute value of the layer
   M = layer .^2;
   [m,n]=size(M);
   w=linspace(0,fs/2,n);                    % Setting the frequency axis
   time=linspace(0,NUM_SAMPLES*T,m);              % Setting the time axis
   clear layer ;                            % Take off the layer matrix from the workspace
   % Ploting the Layer
   % Contour Plots
   figure(1)
   subplot(2,2,yy-3)
   contour(time,w,M')
   title(['"Time - Frequency Plane Layer ',num2str(yy),""])
   xlabel('Time (sec)'),ylabel('Frequency (Hz)'),grid
   colorbar,colormap(jet)
   hold on
   % Color Plots
   figure(2)
   subplot(2,2,yy-3)
   pcolor(time,w,M')
   title(['"Time - Frequency Plane Layer ',num2str(yy),""])
   xlabel('Time (sec)'),ylabel('Frequency (Hz)')
   colorbar,colormap(jet)
   hold on
   % Mesh Plots
   figure(3)
   subplot(2,2,yy-3)
   mesh(time,w,M')
   title(['Cells Energy for Layer',num2str(yy),""])
   xlabel('Time'),ylabel('Frequency'),zlabel('Energy')
   colormap(jet)
   hold on
   % Surf Plots
   figure(4)
   subplot(2,2,yy-3)
   surf(time,w,M')
   title(['Cells Energy for Layer',num2str(yy),""])
   xlabel('Time (sec)'),ylabel('Frequency (Hz)'),zlabel('Energy')
   colormap(jet)
end
%_____________________________________________________________________________________
```

```
%*******************************************************************************
% qmfb.m
%                Quadrature Mirror Filter Bank Function
% qmfb(f,filter,N)
%
% Use:          Quadrature Mirror Filter Bank function.  It takes the input waveform f (formally the signal
%               "tt"), and processes it with a QMF Bank.  'filter' specifies the file containing the filter
%               coefficents of the modified sinc filter to be applyed through the QMFB over the signal.
%               N is an optional input and specifies the last layer to be computed.  This can save
%                time if the higher layers are not needed.  The default is to compute all of the layers.
%
% Author:          Pedro Jarpa. Captain Chilean Air Force.
% Date:            July 30, 2002.
%
%*******************************************************************************
function lay=qmfb(f,filter,N)   % Quadrature Mirror Filter Bank function
%_______________________________________________________________________________
% -- Number of Layers to Compute –
n = floor(log2(length(f)));   % Determine the amount of layers from the length of the signal
if nargin < 3;                    % number of layers
        N = n;
end;
%_______________________________________________________________________________
% -- Formatting the Signal -- (to pass the signal through the filter bank)
I([1:2 .^n],1) = f([1:2 .^n]);
out = I;
%_______________________________________________________________________________
% -- Generating the Output Layers --
% Decompose the function
for lay = 1:N       % layer
 disp(lay)    % Show what layer is been generated
 flag = 1;    % Flag used to set up the columns in the output matrix
 % Reshape the output matrix
 [r,c] = size(out);
 out = zeros(r./2, c.*2);
 for i = 1:2 .^(lay-1)                    % column of I (low to high)
        [G,H] = feval(filter, I(:,i));  % Evaluate the filter (sinc modified) over the signal going by columns
  % Setting the output matrix (Layer) by High Pass (G) and Low Pass (H) filtering output
        if flag
                out(:, i.*2-1) = H;   % Low Pass Branch of QMFB Tree
                out(:, i.*2) = G;     % High Pass Branch of QMFB Tree
        else
                out(:, i.*2-1) = G;   % High Pass Branch of QMFB Tree
                out(:, i.*2) = H;     % Low Pass Branch of QMFB Tree
        end;
                flag = ~flag;          % Change in the flag value
 end;
 % Output Matrix and data to work with
 I = out;           % Output matrix (signal already filtered)
 R = real(I);      % Real part of the filtered signal
 Q = imag(I);    % Imaginary part of the filtered signal
 save(['layer',int2str(lay)],'R','Q');                  % Saving the data to the same directory
%   eval(['save c:\Jarpa\Thesis\data\layer',int2str(lay),'.dat I /ascii /double']);   % Used to save the data in a
different directory
end;
%_______________________________________________________________________________
```

```
%*******************************************************************************
% tsinc.m
%                Modified Sinc Filter Function
% [d1, c1] = tsinc(c0)
%
% Use:          Modified Sinc Filter function. It use truncated modified sinc filter
%                coefficients to decompose the column vector c0 into a (high frequency)
%                column vector d1 and (low frequency) column vector c1.
%                tsinc_su must have been run previously to create h.dat and
%                g.dat files that contain the filter coefficients.
%
% Author:          Pedro Jarpa. Captain Chilean Air Force.
% Date:             July 30, 2002.
%
%*******************************************************************************
function [d1, c1] = tsinc(c0)   % Modified Sinc Filter function
%_____________________________________________________________________________
% -- Load Filter Coefficients (Sinc Modified) --
%
load h.dat;  h = h';
load g.dat;  g = g';
%_____________________________________________________________________________
% -- Setting the Signal --
N = length(c0);                 % Length of the signal (from qmfb function "c0" is a column)
pad3 = length(h)./2 + 1;      % pad with zeros to clear out filter
c0 = [c0; zeros(pad3,1)];    % padding with zeros
%_____________________________________________________________________________
% Decompose the column vector c0
i = pad3:2:(N+pad3-2);           % i will decimate by 2
% c1 low frequency column vector
c1 = filter(fliplr(h),1,c0);         % compute c1
c1 = c1(i);                          % decimate
% d1 high frequency column vector d1
d1 = filter(fliplr(g),1,c0);         % compute d1
d1 = d1(i);                          % decimate
%_____________________________________________________________________________
```

```
%*******************************************************************************
% tsinc_su.m
%               Truncted Sinc Function
% tsinc_su(N)
%
% Use:          Truncated Sinc function finds the coefficients for the truncated sinc modified filters.
%               These are saved in the data directory under h.dat (lpf) and g.dat (hpf).  The Hamming
%               window is used to suppress the Gibb's phenomena.
%               N is the number of coefficients filter pair. For best results, N should be a power of two.
%               If N is 512 and some specific values of S and C in the FIR filter formula yields nearly
%               orthogonal filters with cross correlation of less than 0.001. Due to that the FIR filter found
%               is called "Modified Sinc Filter".
%
% Author:       Pedro Jarpa. Captain Chilean Air Force.
% Date:         July 30, 2002.
%
%*******************************************************************************
function tsinc_su(N);     % Truncted Sinc Function
%_______________________________________________________________________________
% -- Sinc Modified Filter Formula Application --
N = 512;                % Number of filters coefficients pairs.
C = 1.99375872328059;      % Best value to get nearly orthogonal filters with cross correlation of less than
0.001
S = 1.00618488680080;      % Best value to get nearly orthogonal filters with cross correlation of less than
0.001
x = [-floor(N./2):floor(N./2)-1]';  % Vector coefficients
h = sqrt(S./2).*sinc((x + .5)./C);  % Coefficients filter formula
w = hamming(N);                 % Hamming Window
h = w.*h;                       % Low Pass Filter Coefficients
g = h;                          % High Pass Filter Coefficients
g([2:2:N]) = -h([2:2:N]);       % Alternating the sign of the coefficients
%_______________________________________________________________________________
% -- Saving the Data --
save c:\Jarpa\Thesis\data\h.dat h /ascii /double;  % Low Pass Filter Coefficients
save c:\Jarpa\Thesis\data\g.dat g /ascii /double;  % High Pass Filter Coefficients
%_______________________________________________________________________________
```

```
%*******************************************************************************
% multiple_colorplot.m
%                          Time-frequency Plot using “pcolor” command
%
% Use:        Script file to display a color plot of the output layers from the QMFB tree.
%              Needs to write the name of the signal in the “save” command at the end of the code.
%
% Author:         Pedro Jarpa. Captain Chilean Air Force.
% Date:           July 30, 2002.
%
%*******************************************************************************
%_______________________________________________________________________________
% -- Load Layer --
% Write the file directory and layer's number after load command in the next line
yy=0;
counter=1;
for yy=2:lay-1
   load(['c:\Jarpa\Thesis\data\layer',num2str(yy)]) ;  % Setting the file directory and layer's number in the
load command
   % Formating the Layer
   layer = abs(R + j*Q);                          % Taking the absolute value of the layer
   layer = layer/max(max(layer));                 % Normalizing the layer
   M = layer .^2;
   [m,n]=size(M);
   w=linspace(0,fs/2,n);                      % Setting the frequency axis
   time=linspace(0,NUM_SAMPLES*T,m);              % Setting the time axis
   clear layer;                               % Take off the layer matrix from the workspace
   % Color Plots
   figure(counter)
   pcolor(time,w,M')
   title(['"Time - Frequency Plane Layer ',num2str(yy),""])
   xlabel('Time (sec)'),ylabel('Frequency (Hz)')
   colorbar,colormap(jet)
   saveas(gcf,['c:\Jarpa\Thesis\plots\FSK_PSK_T_15_256_10_s_colorplot_layer_',num2str(yy)],'fig');
% CAREFULL!!! ….The name of the signal needs to be changed!!!!!
   counter=counter+1;
end
```

```
%*******************************************************************************
% multiple_contourplot.m
%                          Time-frequency Plot using “contour” command
%
% Use:        Script file to display a contour plot of the output layers from the QMFB tree.
%             Needs to write the name of the signal in the “save” command at the end of the code.
%
% Author:       Pedro Jarpa. Captain Chilean Air Force.
% Date:         July 30, 2002.
%
%*******************************************************************************
%_______________________________________________________________________________
% -- Load Layer --
% Write the file directory and layer's number after load command in the next line
yy=0;
counter=1;
for yy=2:lay-1
   load(['c:\Jarpa\Thesis\data\layer',num2str(yy)]) ;  % Setting the file directory and layer's number in the
load command
   % Formating the Layer
   layer = abs(R + j*Q);                    % Taking the absolute value of the layer
   layer = layer/max(max(layer));           % Normalizing the layer
   M = layer .^2;
   [m,n]=size(M);
   w=linspace(0,fs/2,n);                          % Setting the frequency axis
   time=linspace(0,NUM_SAMPLES*T,m);              % Setting the time axis
   clear layer;                                    % Take off the layer matrix from the workspace
   % Contour Plots
   figure(counter)
   contour(time,w,M')
   title(['"Time - Frequency Plane Layer ',num2str(yy),""])
   xlabel('Time (sec)'),ylabel('Frequency (Hz)'),grid
   colorbar,colormap(jet)
   saveas(gcf,['c:\Jarpa\Thesis\plots\FSK_PSK_T_15_256_10_s_contourplot_layer_',num2str(yy)],'fig');
% CAREFULL...!!...the name of the signal needs to be changed!!!
   counter=counter+1;
end
```

```
%*******************************************************************************
% multiple_ meshplot.m
%                          3 Dimension View Plot using “mesh” command
%
% Use:        Script file to display a mesh plot of the output layers from the QMFB tree.
%              Needs to write the name of the signal in the “save” command at the end of the code.
%
% Author:         Pedro Jarpa. Captain Chilean Air Force.
% Date:           July 30, 2002.
%
%*******************************************************************************
yy=0;
counter=1;
for yy=2:lay-1
   load(['c:\Jarpa\Thesis\data\layer',num2str(yy)]) ;  % Setting the file directory and layer's number in the
load command
   % Formating the Layer
   layer = abs(R + j*Q);                          % Taking the absolute value of the layer
   layer = layer/max(max(layer));                 % Normalizing the layer
   M = layer .^2;
   [m,n]=size(M);
   w=linspace(0,fs/2,n);                                % Setting the frequency axis
   time=linspace(0,NUM_SAMPLES*T,m);                    % Setting the time axis
   clear layer;                                         % Take off the layer matrix from the workspace
   % Mesh Plots
   figure(counter)
%     mesh(time,w,M')
   mesh(time,w,M'),view(90,0)                     % To see the frequency axis
   title(['Cells Energy for Layer',num2str(yy),""])
   xlabel('Time (sec)'),ylabel('Frequency (Hz)'),zlabel('Energy')
   colormap(jet)
   saveas(gcf,['c:\Jarpa\Thesis\plots\FSK_PSK_C_1_15_11_5_0_meshplot_layer_',num2str(yy)],'fig');
% CAREFULL...!!...the name of the signal needs to be changed!!!!!
   counter=counter+1;
end
```

```
%*******************************************************************************
% multiple_ surfplot.m
%                          3 Dimension View Plot using “surf” command
%
% Use:        Script file to display a surf plot of the output layers from the QMFB tree.
%              Needs to write the name of the signal in the “save” command at the end of the code.
%
% Author:         Pedro Jarpa. Captain Chilean Air Force.
% Date:           July 30, 2002.
%
%*******************************************************************************
yy=0;
counter=1;
for yy=2:lay-1
   load(['c:\Jarpa\Thesis\data\layer',num2str(yy)]) ;  % Setting the file directory and layer's number in the
load command
   % Formating the Layer
   layer = abs(R + j*Q);                          % Taking the absolute value of the layer
   layer = layer/max(max(layer));                 % Normalizing the layer
   M = layer .^2;
   [m,n]=size(M);
   w=linspace(0,fs/2,n);                                % Setting the frequency axis
   time=linspace(0,NUM_SAMPLES*T,m);                    % Setting the time axis
   clear layer;                                         % Take off the layer matrix from the workspace
   % Mesh Plots
   figure(counter)
   surf(time,w,M')
%    surf(time,w,M'),view(90,0)                    % To see the frequency axis
   title(['Cells Energy for Layer',num2str(yy),""])
   xlabel('Time (sec)'),ylabel('Frequency (Hz)'),zlabel('Energy')
   colormap(jet)
   saveas(gcf,['c:\Jarpa\Thesis\plots\FSK_PSK_C_1_15_11_5_0_surfplot_layer_',num2str(yy)],'fig');
% CAREFULL...!!...the name of the signal needs to be changed!!!!!
   counter=counter+1;
end
```

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix B: List of Signals Generated by LPIG and Processed by QMFB.

TEST SIGNALS

	Test Signal	Carrier Frequeny	SNR
1	T_1_7_1_s	1000 Hz	Signal Only
2	T_2_7_1_s	1000 Hz, 2000 Hz	Signal Only

BINARY PHASE SHIFT KEYING (BPSK)

	BPSK	Number of bits per Barker code (Bits)	Number of cycles per bit (cpb)	SNR
1	B_1_7_7_1_s	7	1	Signal Only
2	B_1_7_7_1_0	7	1	0 dB
3	B_1_7_7_1_–6	7	1	–6 dB
4	B_1_7_11_1_s	11	1	Signal Only
5	B_1_7_11_1_0	11	1	0 dB
6	B_1_7_11_1_–6	11	1	–6 dB
7	B_1_7_7_5_s	7	5	Signal Only
8	B_1_7_7_5_0	7	5	0 dB
9	B_1_7_7_5_–6	7	5	–6 dB
10	B_1_7_11_5_s	11	5	Signal Only
11	B_1_7_11_5_0	11	5	0 dB
12	B_1_7_11_5_–6	11	5	–6 dB

FREQUENCY MODULATED CONTINUOUS WAVE (FMCW)

	FMCW	Modulation Bandwidth	Modulation Period	SNR
1	F_1_7_250_20_s	250 Hz	20 ms	Signal Only
2	F_1_7_250_20_0	250 Hz	20 ms	0 dB
3	F_1_7_250_20_–6	250 Hz	20 ms	–6 dB
4	F_1_7_250_30_s	500 Hz	30 ms	Signal Only
5	F_1_7_250_30_0	500 Hz	30 ms	0 dB
6	F_1_7_250_30_–6	500 Hz	30 ms	–6 dB
7	F_1_7_500_20_s	250 Hz	20 ms	Signal Only
8	F_1_7_500_20_0	250 Hz	20 ms	0 dB
9	F_1_7_500_20_–6	250 Hz	20 ms	–6 dB
10	F_1_7_500_30_s	500 Hz	30 ms	Signal Only
11	F_1_7_500_30_0	500 Hz	30 ms	0 dB
12	F_1_7_500_30_–6	500 Hz	30 ms	–6 dB

FRANK CODES

	FRANK	Number of code phases N	Number of cycles per phase cpp	SNR
1	FR_1_7_4_1_s	4	1	Signal Only
2	FR_1_7_4_1_0	4	1	0 dB
3	FR_1_7_4_1_–6	4	1	–6 dB
4	FR_1_7_4_5_s	4	5	Signal Only
5	FR_1_7_4_5_0	4	5	0 dB
6	FR_1_7_4_5_–6	4	5	–6 dB
7	FR_1_7_8_1_s	8	1	Signal Only
8	FR_1_7_8_1_0	8	1	0 dB
9	FR_1_7_8_1_–6	8	1	–6 dB
10	FR_1_7_8_5_s	8	5	Signal Only
11	FR_1_7_8_5_0	8	5	0 dB
12	FR_1_7_8_5_–6	8	5	–6 dB

P1 POLYPHASE CODE

	P1	Number of code phases *N*	Number of cycles per phase cpp	SNR
1	P1_1_7_4_1_s	4	1	Signal Only
2	P1_1_7_4_1_0	4	1	0 dB
3	P1_1_7_4_1_–6	4	1	–6 dB
4	P1_1_7_4_5_s	4	5	Signal Only
5	P1_1_7_4_5_0	4	5	0 dB
6	P1_1_7_4_5_–6	4	5	–6 dB
7	P1_1_7_8_1_s	8	1	Signal Only
8	P1_1_7_8_1_0	8	1	0 dB
9	P1_1_7_8_1_–6	8	1	–6 dB
10	P1_1_7_8_5_s	8	5	Signal Only
11	P1_1_7_8_5_0	8	5	0 dB
12	P1_1_7_8_5_–6	8	5	–6 dB

P2 POLYPHASE CODE

	P2	Number of code phases N	Number of cycles per phase cpp	SNR
1	P2_1_7_4_1_s	4	1	Signal Only
2	P2_1_7_4_1_0	4	1	0 dB
3	P2_1_7_4_1_–6	4	1	–6 dB
4	P2_1_7_4_5_s	4	5	Signal Only
5	P2_1_7_4_5_0	4	5	0 dB
6	P2_1_7_4_5_–6	4	5	–6 dB
7	P2_1_7_8_1_s	8	1	Signal Only
8	P2_1_7_8_1_0	8	1	0 dB
9	P2_1_7_8_1_–6	8	1	–6 dB
10	P2_1_7_8_5_s	8	5	Signal Only
11	P2_1_7_8_5_0	8	5	0 dB
12	P2_1_7_8_5_–6	8	5	–6 dB

P3 POLYPHASE CODE

	P3	Number of code phases N^2	Number of cycles per phase cpp	SNR
1	P3_1_7_16_1_s	16	1	Signal Only
2	P3_1_7_16_1_0	16	1	0 dB
3	P3_1_7_16_1_–6	16	1	–6 dB
4	P3_1_7_16_5_s	16	5	Signal Only
5	P3_1_7_16_5_0	16	5	0 dB
6	P3_1_7_16_5_–6	16	5	–6 dB
7	P3_1_7_64_1_s	64	1	Signal Only
8	P3_1_7_64_1_0	64	1	0 dB
9	P3_1_7_64_1_–6	64	1	–6 dB
10	P3_1_7_64_5_s	64	5	Signal Only
11	P3_1_7_64_5_0	64	5	0 dB
12	P3_1_7_64_5_–6	64	5	–6 dB

P4 POLYPHASE CODE

	P4	Number of code phases N^2	Number of cycles per phase cpp	SNR
1	P4_1_7_16_1_s	16	1	Signal Only
2	P4_1_7_16_1_0	16	1	0 dB
3	P4_1_7_16_1_–6	16	1	–6 dB
4	P4_1_7_16_5_s	16	5	Signal Only
5	P4_1_7_16_5_0	16	5	0 dB
6	P4_1_7_16_5_–6	16	5	–6 dB
7	P4_1_7_64_1_s	64	1	Signal Only
8	P4_1_7_64_1_0	64	1	0 dB
9	P4_1_7_64_1_–6	64	1	–6 dB
10	P4_1_7_64_5_s	64	5	Signal Only
11	P4_1_7_64_5_0	64	5	0 dB
12	P4_1_7_64_5_–6	64	5	–6 dB

COSTAS CODE

	COSTAS	Sequence	Number of cycles per phase cpp	SNR
1	C_1_15_10_s	4716523	10	Signal Only
2	C_1_15_10_0	4716523	10	0 dB
3	C_1_15_10_–6	4716523	10	–6 dB
4	C_1_15_20_s	4716523	20	Signal Only
5	C_1_15_20_0	4716523	20	0 dB
6	C_1_15_20_–6	4716523	20	–6 dB
7	C_2_17_10_s	2638751	10	Signal Only
8	C_2_17_10_0	2638751	10	0 dB
9	C_2_17_10_–6	2638751	10	–6 dB
10	C_2_17_20_s	2638751	20	Signal Only
11	C_2_17_20_0	2638751	20	0 dB
12	C_2_17_20_–6	2638751	20	–6 dB

FREQUENCY SHIFT KEYING/PHASE SHIFT KEYING COMBINED WITH COSTAS CODE (FSK/PSK COSTAS)

	FSK/PSK Costas	Sequence	Barker Bits	Number of cycles per bit cpp	SNR
1	FSK_PSK_C_1_15_5_1_s	4716523	5	1	Signal Only
2	FSK_PSK_C_1_15_5_1_0	4716523	5	1	0 dB
3	FSK_PSK_C_1_15_5_5_s	4716523	5	5	Signal Only
4	FSK_PSK_C_1_15_5_5_0	4716523	5	5	0 dB
5	FSK_PSK_C_1_15_11_1_s	4716523	11	1	Signal Only
6	FSK_PSK_C_1_15_11_1_0	4716523	11	1	0 dB
7	FSK_PSK_C_1_15_11_5_s	4716523	11	5	Signal Only
8	FSK_PSK_C_1_15_11_5_0	4716523	11	5	0 dB

FSK/PSK COMBINED WITH TARGET-MATCHED FREQUENCY HOPPING

	FSK/PSK Target	Number of random hops in frequency	Number of cycles and phases	SNR
1	FSK_PSK_T_15_128_5_s	128	5	Signal Only
2	FSK_PSK_T_15_128_5_0	128	5	0 dB
3	FSK_PSK_T_15_128_10_s	128	10	Signal Only
4	FSK_PSK_T_15_128_10_0	128	10	0 dB
5	FSK_PSK_T_15_256_5_s	256	5	Signal Only
6	FSK_PSK_T_15_256_5_0	256	5	0 dB
7	FSK_PSK_T_15_256_10_s	256	10	Signal Only
8	FSK_PSK_T_15_256_10_0	256	10	0 dB

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] Pace, P., Notes for EC4690 (Network Centric Electronic Warfare), Naval Postgraduate School, 2002.

[2] GuoSui, L., Hong, G., WeiMin S., and HongBo, S., "The Analysis and Design of Modern Low Probability of Intercept Radar," *CIE International Conference on Radar Proceedings*, Nanjing University of Science and Technology, Nanjing, China, 2001.

[3] Klein, L., *Millimeter-Wave and Infrared Multisensor Design and Signal Processing*, First Edition, Artech House, Inc, 1997.

[4] Farrell, T. and Prescott, G., **"**A Nine-Tile Algorithm for LPI Signal Detection Using QMF Filter Bank Trees," *MILCOM '96, Conference Proceedings, IEEE* , Vol 3*:* 974-978 (1996).

[5] MATLAB®, "Language of Technical Computing that Integrates Mathematical Computing, Visualization, and a Powerfull Language to Provide a Flexible Enviroment for Technical Computing", *Signal Processing Toolbox User's Guide*, The Mathworks, Natick MA, 1999.

[6] Chui, C., *Wavelets: A Mathematical Tool for Signal Analysis*, First Edition, SIAM, 1997.

[7] Burrus, S., Gopinath, A., and Guo, H., *Introduction to Wavelets and Wavelets Transforms*, First Edition, Prentice Hall, 1998.

[8] Herley, C., Kovacevic, J., Ramchandran, K., and Vetterli, M., "Tilings of the Time-frequency Plane: Construction of Arbitrary Orthogonal Bases and Fast Tiling Algorithms," *IEEE Transactions on Signal Processing,* Vol. 41, No. 12: 3341-3359 (December 1993).

[9] Vetterli, M. and Herley, C., "Wavelets and Filter Banks: Theory and Design." *IEEE Transactions on Signal Processing,* Vol. 40, No. 9: 2207-2232 (September 1992).

[10] Mitra, S., *Digital Signal Processing. A Computer-Based Approach*, Second Edition, McGraw-Hill, 2001.

[11] Proakis, J., and Manolakis, D., *Digital Signal Processing. Principles, Algorithms, and Applications*, Third Edition, Prentice Hall, 1996.

[12] Farrell, T. and Prescott, G., "A Method for Finding Orthogonal Wavelet Filters with Good Energy Tiling Characteristics", *IEEE Transaction on Signal Processing,* Vol. 47, No. 1, (January 1999).

[13] Farrell, T. and Prescott, G., "A Low Probability of Intercept Receiver Using Quadrature Mirror Filter Bank Trees," *ICASSP-96. Conference Proceedings*, Vol 3: 1558-1561 (March 1996).

[14] Personal communication and interchange of information provided by Farrell, T. Wed 3/20/2002 12:19 PM.

[15] Taboada, F., "Detection and Classification of LPI Radar Signals Using Parallel Filter Array and Higher Order Statistics", Master's Thesis, Naval Postgradute School, Monterey, California, September 2002.

[16] Lima, A., "Analysis of LPI Signals Using Cyclostationary Processing", Master's Thesis, Naval Postgraduate School, Monterey, California, September 2002.

[17] Coopeland, D., "A Low Probability of Intercept Signal Detection Receiver Using Quadrature Mirror Filter Bank Trees", *Advance Topics in Electronic Warfare* Course Project, Naval Postgraduate School, June 2001.

[18] Gau, G., "Analysis of Low Probability of Intercept (LPI) Radar Signal using the Wigner Distribution", Master's Thesis, Naval Postgraduate School, Monterey, California, September 2002.

[19] Costas, J., "A Study of a Class of Detection Waveforms Having Nearly Ideal Range-Doppler Ambiguity Properties", *Proceedings of the IEEE*, Vol.72, No 8: 959-966, (August 1984).

[20] Tomasi, W., *Electronic Communications Systems*, Fourth Edition, Prentice Hall, 2001.

[21] Jarpa, P. and Pace, P., "Quantifying the Differences in Low Probability of Intercept Radar Waveforms Using Quadrature Mirror Filtering," Technical Report to be published, Naval Postgraduate School, Monterey, California, 2002.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Chilean Air Attaché
   Chilean Air Force Mission
   Chilean Air Force
   Washington, Washington D.C.

4. Commander of the Division of Education
   Division of Education
   Chilean Air Force
   Santiago, Chile

5. Director of the Aeronautical Polytechnic Academy
   Aeronautical Polytechnic Academy
   Chilean Air Force
   Santiago, Chile

6. Director of Telecommunications and Informatics
   Direction of Telecommunications and Informatics
   Chilean Air Force
   Santiago, Chile

7. Director of Military Polytechnic Academy
   Military Polytechnic Academy
   Chilean Army
   Santiago, Chile

8. Chairman, Code EC
   Electrical and Computer Engineering Department
   Naval Postgraduate School
   Monterey, California

9. Professor Phillip E. Pace, Code EC
   Electrical and Computer Engineering Department
   Naval Postgraduate School
   Monterey, California

10. Captain Pedro Jarpa
Chilean Air Force
Santiago, Chile

11. Professor Herschel H. Loomis, Jr., Code EC
Electrical and Computer Engineering Department
Naval Postgraduate School
Monterey, California

12. Chief of Educational Department
Division of Education
Chilean Air Force
Santiago, Chile